Logtalk APIs

Release v3.103.0

Paulo Moura

Nov 19, 2026

CONTENTS

1 Libraries **1**

1.1 adaptive_boosting_classifier . . . 1
1.1.1 adaptive_boosting_classifier . . . 1
1.2 agglomerative_clusterer . . . 3
1.2.1 agglomerative_clusterer . . . 3
1.3 amqp . . . 4
1.3.1 amqp . . . 4
1.3.2 amqp_pool . . . 23
1.4 anomaly_detection_protocols . . . 28
1.4.1 anomaly_dataset_protocol . . . 28
1.4.2 anomaly_detector_common . . . 30
1.4.3 anomaly_detector_protocol . . . 34
1.5 ant_colony . . . 40
1.5.1 ant_colony(Problem) . . . 40
1.5.2 ant_colony(Problem,Random) . . . 41
1.5.3 ant_colony_problem_protocol . . . 44
1.6 application . . . 47
1.6.1 application_common . . . 47
1.6.2 application_protocol . . . 48
1.7 apriori_pattern_miner . . . 58
1.7.1 apriori_pattern_miner . . . 58
1.8 arbitrary . . . 59
1.8.1 arbitrary . . . 59
1.9 arrangements . . . 65
1.9.1 arrangements . . . 65
1.9.2 arrangements_protocol . . . 66
1.10 asp . . . 74
1.10.1 asp . . . 75
1.10.2 asp_expansion . . . 76
1.10.3 asp_output . . . 77
1.10.4 asp_problem_protocol . . . 79
1.10.5 asp_process . . . 80
1.10.6 asp_program . . . 83
1.10.7 asp_protocol . . . 86
1.10.8 asp_symbols . . . 93
1.11 assertions . . . 94
1.11.1 assertions . . . 94
1.11.2 assertions(Mode) . . . 95
1.11.3 assertions_messages . . . 97
1.12 assignvars . . . 98
1.12.1 assignvars . . . 98
1.12.2 assignvarsp . . . 99

1.13 association_rule_miner . . . 102
1.13.1 association_rule_miner . . . 102
1.13.2 association_rule_miner_common . . . 103
1.13.3 association_rule_miner_protocol . . . 105
1.14 atms . . . 110
1.14.1 atms . . . 110
1.14.2 atms_bitset_environment . . . 111
1.14.3 atms_environment_protocol . . . 112
1.14.4 atms_ordered_list_environment . . . 115
1.14.5 atms_protocol . . . 116
1.14.6 atms_segmented_bitset_environment . . . 126
1.15 avro . . . 127
1.15.1 avro . . . 127
1.16 base32 . . . 130
1.16.1 base32 . . . 130
1.17 base58 . . . 132
1.17.1 base58 . . . 132
1.18 base64 . . . 134
1.18.1 base64 . . . 134
1.18.2 base64url . . . 136
1.18.3 base64url_no_padding . . . 138
1.19 base85 . . . 140
1.19.1 base85 . . . 140
1.20 baseline_forecasting . . . 142
1.20.1 baseline_forecasting . . . 142
1.21 bayesian_ridge_regression . . . 143
1.21.1 bayesian_ridge_regression . . . 143
1.22 block_ciphers . . . 145
1.22.1 aes128 . . . 145
1.22.2 aes192 . . . 146
1.22.3 aes256 . . . 147
1.22.4 aes_common(KeySize,Nk,Nr) . . . 148
1.22.5 block_cipher_modes_common . . . 150
1.22.6 block_cipher_prepared_key_protocol . . . 154
1.22.7 block_cipher_protocol . . . 157
1.22.8 cbc . . . 160
1.22.9 ctr . . . 163
1.22.10 ecb . . . 166
1.23 bm25_recommender . . . 169
1.23.1 bm25_recommender . . . 169
1.24 borda_ranker . . . 186
1.24.1 borda_ranker . . . 186
1.25 bradley_terry_ranker . . . 188
1.25.1 bradley_terry_ranker . . . 188
1.26 bson . . . 190
1.26.1 bson . . . 190
1.26.2 bson(StringRepresentation) . . . 191
1.27 byte_order . . . 193
1.27.1 byte_order . . . 193
1.28 c45_classifier . . . 196
1.28.1 c45_classifier . . . 197
1.29 cartesian_products . . . 200
1.29.1 cartesian_products . . . 200
1.29.2 cartesian_products_protocol . . . 201
1.30 cbor . . . 209
1.30.1 cbor . . . 210

1.30.2 cbor(StringRepresentation) . . . 211
1.31 ccsds_frames . . . 212
1.31.1 ccsds_aos_frames(FrameLength,InsertZoneLength,HasOCF,HasFECF) . . . 213
1.31.2 ccsds_frame_protocol . . . 216
1.31.3 ccsds_frames . . . 221
1.31.4 ccsds_frames_types . . . 227
1.31.5 ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF) . . . 228
1.31.6 ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF) . . . 231
1.32 ccsds_link_profiles . . . 236
1.32.1 ccsds_link_profiles . . . 236
1.33 ccsds_packet_services . . . 247
1.33.1 ccsds_packet_services . . . 247
1.34 ccsds_packetization . . . 267
1.34.1 ccsds_packetization . . . 267
1.35 ccsds_packets . . . 275
1.35.1 ccsds_packets . . . 275
1.35.2 ccsds_packets(SecondaryHeaderLength) . . . 276
1.35.3 ccsds_packets_types . . . 282
1.36 ccsds_tc_services . . . 283
1.36.1 ccsds_tc_services . . . 283
1.37 ccsds_time_codes . . . 298
1.37.1 ccsds_ccs(CalendarVariant,FractionOctets) . . . 298
1.37.2 ccsds_cds(DaySegmentOctets,SubmillisecondOctets,Epoch) . . . 300
1.37.3 ccsds_cuc(CoarseOctets,FineOctets,Epoch) . . . 302
1.37.4 ccsds_time_code_protocol . . . 304
1.37.5 ccsds_time_codes . . . 308
1.37.6 ccsds_time_codes_types . . . 310
1.38 ccsds_time_fields . . . 311
1.38.1 ccsds_time_fields . . . 311
1.39 character_sets . . . 314
1.39.1 character_set . . . 314
1.39.2 character_set_protocol . . . 317
1.39.3 iso_8859_1 . . . 320
1.39.4 iso_8859_10 . . . 321
1.39.5 iso_8859_13 . . . 322
1.39.6 iso_8859_14 . . . 323
1.39.7 iso_8859_15 . . . 324
1.39.8 iso_8859_16 . . . 325
1.39.9 iso_8859_2 . . . 326
1.39.10 iso_8859_3 . . . 327
1.39.11 iso_8859_4 . . . 328
1.39.12 iso_8859_9 . . . 329
1.39.13 mapped_single_byte_character_set . . . 330
1.39.14 single_byte_character_set(MaxCode) . . . 331
1.39.15 us_ascii . . . 332
1.39.16 utf_16_character_set(Endian) . . . 333
1.39.17 utf_16be . . . 334
1.39.18 utf_16le . . . 335
1.39.19 utf_32_character_set(Endian) . . . 336
1.39.20 utf_32be . . . 337
1.39.21 utf_32le . . . 338
1.39.22 utf_8 . . . 339
1.39.23 utf_8_character_set . . . 340
1.39.24 windows_1250 . . . 341
1.39.25 windows_1251 . . . 342
1.39.26 windows_1252 . . . 343

1.39.27 windows_1253 . . . 344
1.39.28 windows_1254 . . . 345
1.39.29 windows_1257 . . . 346
1.40 chi_square_feature_selector . . . 347
1.40.1 chi_square_feature_selector . . . 347
1.41 classification_protocols . . . 349
1.41.1 classifier_common . . . 349
1.41.2 classifier_protocol . . . 358
1.41.3 dataset_protocol . . . 363
1.41.4 probabilistic_classifier_common . . . 365
1.41.5 probabilistic_classifier_protocol . . . 367
1.42 clo_span_pattern_miner . . . 368
1.42.1 clo_span_pattern_miner . . . 368
1.43 clustering_protocols . . . 369
1.43.1 clusterer_common . . . 369
1.43.2 clusterer_protocol . . . 380
1.43.3 clustering_dataset_protocol . . . 385
1.43.4 search_indexing . . . 387
1.44 cmac . . . 390
1.44.1 cmac . . . 390
1.44.2 cmac_protocol . . . 392
1.45 code_metrics . . . 395
1.45.1 cc_metric . . . 395
1.45.2 code_metric . . . 396
1.45.3 code_metrics . . . 407
1.45.4 code_metrics_messages . . . 408
1.45.5 code_metrics_utilities . . . 409
1.45.6 cogc_metric . . . 414
1.45.7 coupling_metric . . . 415
1.45.8 dit_metric . . . 416
1.45.9 doc_metric . . . 418
1.45.10 halstead_metric . . . 421
1.45.11 halstead_metric(Stroud) . . . 422
1.45.12 lcom_metric . . . 424
1.45.13 lines_metric . . . 425
1.45.14 mi_metric . . . 426
1.45.15 mi_metric(Stroud) . . . 427
1.45.16 noc_metric . . . 428
1.45.17 nor_metric . . . 429
1.45.18 rfc_metric . . . 431
1.45.19 size_metric . . . 432
1.45.20 upn_metric . . . 433
1.45.21 wmc_metric . . . 434
1.46 colley_ranker . . . 435
1.46.1 colley_ranker . . . 436
1.47 combinations . . . 437
1.47.1 combinations . . . 437
1.47.2 combinations_protocol . . . 438
1.48 command_line_options . . . 447
1.48.1 command_line_option . . . 447
1.48.2 command_line_options . . . 451
1.49 constrained_optimization . . . 456
1.49.1 augmented_lagrangian(Problem,InnerSolver) . . . 456
1.49.2 barrier_sub_problem(Problem,OuterIteration,StartPoint,Mu,Rho,Sign) . . . 457
1.49.3 constrained_optimization_problem_protocol . . . 459
1.49.4 constrained_optimization_solver . . . 462

1.49.5 log_barrier(Problem,InnerSolver) . . . 466
1.49.6 penalty_sub_problem(Problem,OuterIteration,StartPoint,Rho,Sign) . . . 467
1.49.7 phase1_sub_problem(Problem,StartPoint) . . . 468
1.49.8 primal_dual_interior_point(Problem) . . . 469
1.49.9 qp_active_set . . . 471
1.49.10 qp_solver_protocol . . . 472
1.49.11 quadratic_penalty(Problem,InnerSolver) . . . 473
1.49.12 sqp_active_set(Problem) . . . 475
1.49.13 sub_problem(Problem,OuterIteration,StartPoint,EqLambda,IneqLambda, Rho,Sign) . . . 476
1.50 copeland_ranker . . . 477
1.50.1 copeland_ranker . . . 477
1.51 core . . . 479
1.51.1 core_messages . . . 479
1.51.2 expanding . . . 480
1.51.3 forwarding . . . 482
1.51.4 logtalk . . . 483
1.51.5 monitoring . . . 496
1.51.6 user . . . 498
1.52 coroutining . . . 499
1.52.1 coroutining . . . 499
1.53 crontab . . . 502
1.53.1 crontab . . . 502
1.53.2 crontab(Representation) . . . 503
1.54 crs_projections . . . 506
1.54.1 crs_projections . . . 507
1.54.2 crs_projections_protocol . . . 508
1.55 crypto . . . 514
1.55.1 authenticated_channel . . . 514
1.55.2 crypto . . . 520
1.55.3 ed25519 . . . 535
1.55.4 x25519 . . . 538
1.55.5 xchacha20_poly1305 . . . 540
1.56 csv . . . 545
1.56.1 csv . . . 545
1.56.2 csv(Header,Separator,IgnoreQuotes) . . . 546
1.56.3 csv(Header,Separator,IgnoreQuotes,Comments) . . . 547
1.56.4 csv_guess_questions . . . 548
1.56.5 csv_protocol . . . 549
1.57 cuid2 . . . 558
1.57.1 cuid2 . . . 558
1.57.2 cuid2(Representation,Size,Alphabet) . . . 559
1.57.3 cuid2_protocol . . . 561
1.58 cusum_anomaly_detector . . . 562
1.58.1 cusum_anomaly_detector . . . 562
1.59 datalog . . . 563
1.59.1 datalog . . . 563
1.59.2 datalog_protocol . . . 572
1.60 dates . . . 578
1.60.1 date . . . 578
1.60.2 datep . . . 579
1.60.3 time . . . 589
1.60.4 timep . . . 590
1.61 dates_tz . . . 592
1.61.1 dates_tz . . . 592
1.61.2 dates_tz_protocol . . . 593

1.62 dbscan_clusterer . . . 596
1.62.1 dbscan_clusterer . . . 596
1.63 dead_code_scanner . . . 598
1.63.1 dead_code_scanner . . . 598
1.63.2 dead_code_scanner_messages . . . 605
1.64 debug_messages . . . 606
1.64.1 debug_messages . . . 607
1.65 debugger . . . 610
1.65.1 debugger . . . 610
1.65.2 debugger_messages . . . 618
1.65.3 debuggerp . . . 619
1.65.4 dump_trace . . . 629
1.66 dependents . . . 631
1.66.1 observer . . . 631
1.66.2 subject . . . 632
1.67 deques . . . 635
1.67.1 deque . . . 635
1.67.2 deque_protocol . . . 636
1.68 derangements . . . 641
1.68.1 derangements . . . 642
1.68.2 derangements_protocol . . . 643
1.69 diagrams . . . 652
1.69.1 caller_diagram . . . 652
1.69.2 caller_diagram(Format) . . . 653
1.69.3 cytoscapejs_graph_language . . . 655
1.69.4 d2_graph_language . . . 658
1.69.5 diagram(Format) . . . 660
1.69.6 diagrams . . . 680
1.69.7 diagrams(Format) . . . 681
1.69.8 directory_dependency_diagram . . . 689
1.69.9 directory_dependency_diagram(Format) . . . 690
1.69.10 directory_diagram(Format) . . . 692
1.69.11 directory_load_diagram . . . 695
1.69.12 directory_load_diagram(Format) . . . 696
1.69.13 dot_graph_language . . . 697
1.69.14 entity_diagram . . . 699
1.69.15 entity_diagram(Format) . . . 700
1.69.16 file_dependency_diagram . . . 703
1.69.17 file_dependency_diagram(Format) . . . 704
1.69.18 file_diagram(Format) . . . 706
1.69.19 file_load_diagram . . . 709
1.69.20 file_load_diagram(Format) . . . 710
1.69.21 graph_language_protocol . . . 711
1.69.22 graph_language_registry . . . 715
1.69.23 inheritance_diagram . . . 716
1.69.24 inheritance_diagram(Format) . . . 717
1.69.25 library_dependency_diagram . . . 718
1.69.26 library_dependency_diagram(Format) . . . 720
1.69.27 library_diagram(Format) . . . 721
1.69.28 library_load_diagram . . . 725
1.69.29 library_load_diagram(Format) . . . 726
1.69.30 mermaid_graph_language . . . 728
1.69.31 modules_diagram_support . . . 729
1.69.32 plantuml_graph_language . . . 731
1.69.33 uses_diagram . . . 732
1.69.34 uses_diagram(Format) . . . 733

1.69.35 xref_diagram . . . 735
1.69.36 xref_diagram(Format) . . . 736
1.70 dictionaries . . . 739
1.70.1 avltree . . . 739
1.70.2 bintree . . . 740
1.70.3 dictionaryp . . . 742
1.70.4 rbtree . . . 753
1.70.5 splaytree . . . 754
1.70.6 two3tree . . . 756
1.71 dif . . . 767
1.71.1 dif . . . 767
1.72 differential_evolution . . . 769
1.72.1 differential_evolution(Problem) . . . 769
1.72.2 differential_evolution(Problem,Random) . . . 770
1.72.3 differential_evolution_problem_protocol . . . 774
1.73 dimension_reduction_protocols . . . 776
1.73.1 dimension_reducer_common . . . 776
1.73.2 dimension_reducer_protocol . . . 788
1.73.3 dimension_reduction_dataset_protocol . . . 793
1.73.4 supervised_dimension_reduction_dataset_protocol . . . 795
1.73.5 target_supervised_dimension_reduction_dataset_protocol . . . 796
1.74 doclet . . . 798
1.74.1 doclet . . . 798
1.75 eclat_pattern_miner . . . 800
1.75.1 eclat_pattern_miner . . . 800
1.76 edcg . . . 801
1.76.1 edcg . . . 801
1.77 elastic_net_regression . . . 804
1.77.1 elastic_net_regression . . . 804
1.78 elo_ranker . . . 805
1.78.1 elo_ranker . . . 805
1.79 emeans_clusterer . . . 807
1.79.1 emeans_clusterer . . . 807
1.79.2 emeans_clusterer(Random) . . . 808
1.80 events . . . 809
1.80.1 after_event_registry . . . 810
1.80.2 before_event_registry . . . 811
1.80.3 event_registry . . . 812
1.80.4 event_registryp . . . 813
1.80.5 monitor . . . 816
1.80.6 monitorp . . . 817
1.81 ewma_anomaly_detector . . . 820
1.81.1 ewma_anomaly_detector . . . 820
1.82 expand_library_alias_paths . . . 821
1.82.1 expand_library_alias_paths . . . 821
1.83 expecteds . . . 822
1.83.1 either . . . 823
1.83.2 expected . . . 826
1.83.3 expected(Expected) . . . 830
1.84 exponential_smoothing . . . 840
1.84.1 exponential_smoothing . . . 840
1.84.2 exponential_smoothing_common . . . 843
1.84.3 exponential_smoothing_problem(Method,Series,OptimizationSeries, Frequency,ParameterSpecification,InitializationSpecification) . . . 848
1.85 fcube . . . 849
1.85.1 fcube . . . 849

1.86 feature_selection_protocols . . . 852
1.86.1 anova_f_score . . . 852
1.86.2 chi_square_score . . . 853
1.86.3 chi_square_score(Discretization) . . . 854
1.86.4 chi_square_yates_score . . . 855
1.86.5 correlation_score . . . 856
1.86.6 cramers_v_bias_corrected_score . . . 857
1.86.7 cramers_v_bias_corrected_score(Discretization) . . . 858
1.86.8 cramers_v_score . . . 859
1.86.9 cramers_v_score(Discretization) . . . 860
1.86.10 feature_dataset_protocol . . . 861
1.86.11 feature_discretization . . . 863
1.86.12 feature_redundancy . . . 865
1.86.13 feature_scoring_common . . . 866
1.86.14 feature_scoring_protocol . . . 875
1.86.15 feature_selector_common . . . 876
1.86.16 feature_selector_protocol . . . 884
1.86.17 filter_feature_selector_common . . . 891
1.86.18 fisher_score . . . 895
1.86.19 mutual_information_score . . . 896
1.86.20 mutual_information_score(Discretization) . . . 897
1.86.21 relief_feature_selector_common . . . 898
1.86.22 symmetrical_uncertainty_score . . . 902
1.86.23 symmetrical_uncertainty_score(Discretization) . . . 903
1.86.24 variance_score . . . 904
1.87 fisher_score_feature_selector . . . 905
1.87.1 fisher_score_feature_selector . . . 905
1.88 flags . . . 906
1.88.1 flags . . . 907
1.88.2 flags_validator . . . 914
1.89 flow_networks . . . 915
1.89.1 capacity_scaling . . . 915
1.89.2 capacity_scaling(Dictionary) . . . 916
1.89.3 dinic . . . 917
1.89.4 dinic(Dictionary) . . . 918
1.89.5 edmonds_karp . . . 919
1.89.6 edmonds_karp(Dictionary) . . . 920
1.89.7 flow_network . . . 922
1.89.8 flow_network(Dictionary) . . . 923
1.89.9 flow_network_protocol . . . 924
1.89.10 flow_support(Dictionary) . . . 931
1.89.11 gomory_hu . . . 936
1.89.12 gomory_hu(Dictionary) . . . 937
1.89.13 max_flow_common(Dictionary) . . . 939
1.89.14 max_flow_protocol . . . 941
1.89.15 min_cost_flow_protocol . . . 946
1.89.16 piecewise_costs . . . 949
1.89.17 piecewise_costs(Dictionary) . . . 950
1.89.18 successive_shortest_paths . . . 953
1.89.19 successive_shortest_paths(Dictionary) . . . 954
1.89.20 time_expanded_network . . . 955
1.89.21 time_expanded_network(Dictionary) . . . 956
1.90 format . . . 958
1.90.1 format . . . 958
1.91 fp_growth_pattern_miner . . . 960
1.91.1 fp_growth_pattern_miner . . . 960

1.92 frequent_pattern_mining_protocols . . . 961
1.92.1 frequent_pattern_mining_common . . . 961
1.92.2 transaction_dataset_protocol . . . 964
1.93 gaussian_mixture_clusterer . . . 966
1.93.1 gaussian_mixture_clusterer . . . 966
1.94 gaussian_process_regression . . . 967
1.94.1 gaussian_process_regression . . . 967
1.95 genetic_algorithm . . . 969
1.95.1 genetic_algorithm(Problem) . . . 969
1.95.2 genetic_algorithm(Problem,Random) . . . 970
1.95.3 genetic_algorithm_problem_protocol . . . 973
1.96 genint . . . 978
1.96.1 genint . . . 978
1.96.2 genint_core . . . 979
1.97 gensym . . . 981
1.97.1 gensym . . . 981
1.97.2 gensym_core . . . 982
1.98 geohash . . . 984
1.98.1 geohash . . . 984
1.98.2 geohash_protocol . . . 986
1.99 geojson . . . 998
1.99.1 geojson . . . 998
1.99.2 geojson(ObjectRepresentation,PairRepresentation,StringRepresentation) 999
1.99.3 geojson_protocol . . . 1000
1.100 geospatial . . . 1005
1.100.1 geospatial . . . 1005
1.100.2 geospatial_protocol . . . 1006
1.101 git . . . 1028
1.101.1 git . . . 1028
1.101.2 git_protocol . . . 1029
1.102 glicko2_periodic_ranker . . . 1034
1.102.1 glicko2_periodic_ranker . . . 1034
1.103 glicko2_ranker . . . 1036
1.103.1 glicko2_ranker . . . 1036
1.104 gpx . . . 1037
1.104.1 gpx . . . 1037
1.104.2 gpx_protocol . . . 1038
1.105 gradient_boosting_classifier . . . 1042
1.105.1 gradient_boosting_classifier . . . 1042
1.106 gradient_boosting_regression . . . 1044
1.106.1 gradient_boosting_regression . . . 1044
1.107 grammars . . . 1045
1.107.1 blank_grammars(Format) . . . 1045
1.107.2 ip_grammars(Format) . . . 1050
1.107.3 number_grammars(Format) . . . 1052
1.107.4 sequence_grammars . . . 1057
1.108 graphs . . . 1060
1.108.1 directed_graph_common . . . 1060
1.108.2 directed_graph_protocol . . . 1061
1.108.3 graph_common . . . 1066
1.108.4 graph_protocol . . . 1068
1.108.5 graph_types . . . 1079
1.108.6 undirected_graph_common . . . 1080
1.108.7 unweighted_directed_graph . . . 1085
1.108.8 unweighted_directed_graph(Dictionary) . . . 1086
1.108.9 unweighted_graph_common(Dictionary) . . . 1088

1.108.10 unweighted_graph_protocol . . . 1090
1.108.11 unweighted_undirected_graph . . . 1092
1.108.12 unweighted_undirected_graph(Dictionary) . . . 1093
1.108.13 weighted_directed_graph . . . 1095
1.108.14 weighted_directed_graph(Dictionary) . . . 1096
1.108.15 weighted_graph_common(Dictionary) . . . 1097
1.108.16 weighted_graph_protocol . . . 1100
1.108.17 weighted_undirected_graph . . . 1103
1.108.18 weighted_undirected_graph(Dictionary) . . . 1105
1.109 gravatar . . . 1107
1.109.1 gravatar . . . 1107
1.109.2 gravatar_api . . . 1115
1.110 gsp_pattern_miner . . . 1117
1.110.1 gsp_pattern_miner . . . 1117
1.111 hashes . . . 1118
1.111.1 blake2_core(Key,DigestSize) . . . 1118
1.111.2 blake2b . . . 1123
1.111.3 blake2b(Key,DigestSize) . . . 1124
1.111.4 blake2s . . . 1125
1.111.5 blake2s(Key,DigestSize) . . . 1126
1.111.6 crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength) . . . 1127
1.111.7 crc32_reflected(Polynomial) . . . 1129
1.111.8 crc32b . . . 1130
1.111.9 crc32bzip2 . . . 1131
1.111.10 crc32c . . . 1132
1.111.11 crc32mpeg2 . . . 1133
1.111.12 crc32posix . . . 1134
1.111.13 crc32q . . . 1135
1.111.14 djb2_32 . . . 1136
1.111.15 djb2_64 . . . 1137
1.111.16 fips202_fixed_hash(RateBytes,Suffix,OutputBytes) . . . 1138
1.111.17 fips202_hash(RateBytes,Suffix,OutputBytes) . . . 1139
1.111.18 fnv1a_32 . . . 1141
1.111.19 fnv1a_64 . . . 1142
1.111.20 hash_common_32 . . . 1143
1.111.21 hash_common_64 . . . 1148
1.111.22 hash_digest_protocol . . . 1153
1.111.23 hash_protocol . . . 1155
1.111.24 hash_state_protocol . . . 1157
1.111.25 md5 . . . 1159
1.111.26 murmurhash3_x64_128 . . . 1161
1.111.27 murmurhash3_x86_128 . . . 1162
1.111.28 murmurhash3_x86_32 . . . 1163
1.111.29 sdbm_32 . . . 1164
1.111.30 sdbm_64 . . . 1165
1.111.31 sha1 . . . 1166
1.111.32 sha224 . . . 1167
1.111.33 sha256 . . . 1168
1.111.34 sha384 . . . 1170
1.111.35 sha3_224 . . . 1171
1.111.36 sha3_256 . . . 1172
1.111.37 sha3_384 . . . 1173
1.111.38 sha3_512 . . . 1174
1.111.39 sha512 . . . 1175
1.111.40 sha512_256 . . . 1176
1.111.41 shake128(OutputBytes) . . . 1177

1.111.42 shake256(OutputBytes) . . . 1178
1.111.43 siphash_2_4 . . . 1179
1.111.44 siphash_2_4(Key) . . . 1180
1.112 hdbscan_clusterer . . . 1181
1.112.1 hdbscan_clusterer . . . 1182
1.113 heaps . . . 1183
1.113.1 binary_heap(Order) . . . 1183
1.113.2 binary_heap_max . . . 1184
1.113.3 binary_heap_min . . . 1185
1.113.4 heap(Order) . . . 1186
1.113.5 heap_protocol . . . 1187
1.113.6 heapp . . . 1192
1.113.7 maxheap . . . 1193
1.113.8 minheap . . . 1194
1.113.9 pairing_heap(Order) . . . 1195
1.113.10 pairing_heap_max . . . 1196
1.113.11 pairing_heap_min . . . 1197
1.114 help . . . 1198
1.114.1 help . . . 1198
1.115 hierarchical_clustering . . . 1205
1.115.1 hierarchical_clustering . . . 1206
1.116 hierarchies . . . 1207
1.116.1 class_hierarchy . . . 1207
1.116.2 class_hierarchyp . . . 1208
1.116.3 hierarchyp . . . 1215
1.116.4 proto_hierarchy . . . 1218
1.116.5 proto_hierarchyp . . . 1219
1.117 hmac . . . 1221
1.117.1 hmac . . . 1221
1.117.2 hmac_protocol . . . 1222
1.118 hodge_rank . . . 1225
1.118.1 hodge_rank . . . 1225
1.119 hook_flows . . . 1226
1.119.1 hook_pipeline(Pipeline) . . . 1227
1.119.2 hook_set(Set) . . . 1228
1.120 hook_objects . . . 1229
1.120.1 backend_adapter_hook . . . 1229
1.120.2 default_workflow_hook . . . 1230
1.120.3 grammar_rules_hook . . . 1231
1.120.4 identity_hook . . . 1232
1.120.5 object_wrapper_hook . . . 1233
1.120.6 object_wrapper_hook(Protocol) . . . 1234
1.120.7 object_wrapper_hook(Name,Relations) . . . 1235
1.120.8 print_goal_hook . . . 1237
1.120.9 prolog_module_hook(Module) . . . 1238
1.120.10 suppress_goal_hook . . . 1239
1.120.11 write_to_file_hook(File) . . . 1240
1.120.12 write_to_file_hook(File,Options) . . . 1241
1.120.13 write_to_stream_hook(Stream) . . . 1242
1.120.14 write_to_stream_hook(Stream,Options) . . . 1243
1.121 html . . . 1244
1.121.1 html . . . 1244
1.121.2 html5 . . . 1247
1.121.3 xhtml11 . . . 1248
1.122 http_authenticate . . . 1249
1.122.1 http_authenticate . . . 1249

1.122.2 http_authenticate_verifier_protocol . . . 1255
1.122.3 http_htpasswd_verifier(Path) . . . 1257
1.122.4 http_router_basic_auth(Verifier,BaseOptions) . . . 1258
1.122.5 http_server_core_basic_handler(Verifier,Handler,ProtectOptions) . . . 1260
1.123 http_client . . . 1261
1.123.1 http_client . . . 1261
1.124 http_client_core . . . 1275
1.124.1 http_client_core . . . 1275
1.125 http_cookies . . . 1279
1.125.1 http_cookies(Representation) . . . 1279
1.126 http_core . . . 1284
1.126.1 http_authentication_helpers . . . 1285
1.126.2 http_body_codec_protocol . . . 1286
1.126.3 http_core . . . 1289
1.126.4 http_docroot_paths . . . 1304
1.126.5 http_form_body_codec . . . 1307
1.126.6 http_handler_protocol . . . 1308
1.126.7 http_json_body_codec . . . 1309
1.126.8 http_json_term_helpers . . . 1310
1.126.9 http_message_helpers . . . 1314
1.126.10 http_octet_stream_body_codec . . . 1317
1.126.11 http_origin_site_helpers . . . 1318
1.126.12 http_request_protocol . . . 1321
1.126.13 http_response_protocol . . . 1324
1.126.14 http_text_body_codec . . . 1327
1.126.15 http_text_helpers . . . 1328
1.127 http_cors . . . 1331
1.127.1 http_cors . . . 1331
1.128 http_digest . . . 1335
1.128.1 http_client_digest_session . . . 1335
1.128.2 http_client_digest_session(HTTPTransport) . . . 1336
1.128.3 http_digest . . . 1344
1.128.4 http_digest_verifier_protocol . . . 1354
1.128.5 http_router_digest_auth(Verifier,ProtectOptions, AuthenticationInfoOptions) . . . 1355
1.128.6 http_server_core_digest_handler(Verifier,Handler,ProtectOptions, AuthenticationInfoOptions) . . . 1358
1.129 http_directory_listing . . . 1359
1.129.1 http_directory_listing . . . 1360
1.130 http_htmx . . . 1362
1.130.1 http_htmx . . . 1362
1.130.2 http_router_htmx . . . 1371
1.131 http_multipart . . . 1373
1.131.1 http_multipart . . . 1373
1.132 http_oauth . . . 1381
1.132.1 http_oauth . . . 1381
1.132.2 http_oauth_exact_scope_checker . . . 1393
1.132.3 http_oauth_helpers . . . 1394
1.132.4 http_oauth_introspection_client . . . 1397
1.132.5 http_oauth_introspection_verifier(Endpoint,Options) . . . 1399
1.132.6 http_oauth_jwt_verifier(KeyOrJWKSet,Options) . . . 1400
1.132.7 http_oauth_metadata . . . 1401
1.132.8 http_oauth_scope_checker_protocol . . . 1405
1.132.9 http_oauth_verifier_protocol . . . 1406
1.132.10 http_router_oauth(Verifier,BaseOptions) . . . 1408

1.132.11 http_server_core_oauth_endpoint_handler(ProtectedResource, MetadataDescriptors,MetadataOptions,Verifier,Handler,ProtectOptions) . 1410
1.132.12 http_server_core_oauth_handler(Verifier,Handler,ProtectOptions) 1411
1.133 http_parameters 1412
1.133.1 http_parameters 1412
1.133.2 http_router_parameters 1416
1.134 http_process_transport 1418
1.134.1 http_process_transport 1418
1.135 http_router 1426
1.135.1 http_router 1426
1.136 http_server 1436
1.136.1 http_server 1437
1.137 http_server_core 1451
1.137.1 http_server_core 1451
1.138 http_session 1457
1.138.1 http_client_session 1457
1.138.2 http_client_session(HTTPTransport) 1458
1.138.3 http_cookie_jar 1466
1.138.4 http_router_server_session(Manager) 1474
1.138.5 http_server_core_session 1476
1.138.6 http_server_core_session_handler(Manager,Handler) 1486
1.139 http_socket_transport 1487
1.139.1 http_socket_transport 1487
1.139.2 http_transport_protocol 1491
1.140 http_sse 1507
1.140.1 http_sse 1507
1.140.2 http_sse_accept_handler(Options) 1523
1.140.3 http_sse_service_handler_protocol 1524
1.141 http_static_files 1526
1.141.1 http_static_files 1526
1.142 http_websocket 1528
1.142.1 http_websocket 1528
1.142.2 http_websocket_accept_handler(Options) 1541
1.143 http_websocket_frames 1542
1.143.1 http_websocket_frames 1542
1.144 http_websocket_handshake 1549
1.144.1 http_websocket_handshake 1549
1.145 http_websocket_messages 1551
1.145.1 http_websocket_messages 1551
1.145.2 http_websocket_messages(TextRepresentation) 1552
1.146 http_websocket_service 1557
1.146.1 http_websocket_client_service 1557
1.146.2 http_websocket_client_service(HTTPTransport) 1558
1.146.3 http_websocket_server_service 1562
1.146.4 http_websocket_server_service(HTTPTransport) 1563
1.146.5 http_websocket_service(HTTPTransport,Role,TextRepresentation) 1570
1.146.6 http_websocket_service_handler_protocol 1575
1.146.7 http_websocket_service_loop_options 1577
1.146.8 http_websocket_service_registry 1578
1.147 http_websocket_session 1585
1.147.1 http_websocket_client_session 1585
1.147.2 http_websocket_server_session 1586
1.147.3 http_websocket_session(Role,TextRepresentation) 1587
1.147.4 http_websocket_session_read_options 1597
1.147.5 http_websocket_session_write_options 1598
1.148 ica_projection 1600

1.148.1 ica_projection . . . 1600
1.149 ids . . . 1601
1.149.1 ids . . . 1601
1.149.2 ids(Representation,Bytes) . . . 1602
1.150 ieee_754 . . . 1604
1.150.1 ieee_754(Precision,ByteOrder,NaNRepresentation) . . . 1604
1.150.2 ieee_754_common(Precision,ByteOrder) . . . 1605
1.150.3 ieee_754_fields(Precision,ByteOrder) . . . 1610
1.150.4 ieee_754_fields_protocol . . . 1611
1.150.5 ieee_754_protocol . . . 1615
1.151 integer_partitions . . . 1620
1.151.1 integer_partitions . . . 1620
1.151.2 integer_partitions_protocol . . . 1621
1.152 intermittent_demand_forecasting . . . 1637
1.152.1 intermittent_demand_forecasting . . . 1638
1.153 intervals . . . 1652
1.153.1 interval . . . 1652
1.153.2 interval_algebra . . . 1653
1.153.3 interval_algebra_protocol . . . 1654
1.153.4 interval_constraint_network . . . 1656
1.153.5 interval_constraint_network_protocol . . . 1657
1.153.6 interval_protocol . . . 1667
1.153.7 interval_relation_set . . . 1673
1.153.8 interval_relation_set_protocol . . . 1674
1.154 iqr_anomaly_detector . . . 1679
1.154.1 iqr_anomaly_detector . . . 1679
1.155 iso8601 . . . 1680
1.155.1 iso8601 . . . 1680
1.156 iso_13616 . . . 1693
1.156.1 iso_13616 . . . 1693
1.156.2 iso_13616_protocol . . . 1694
1.156.3 iso_13616_registry . . . 1696
1.157 iso_3166 . . . 1697
1.157.1 iso_3166 . . . 1697
1.157.2 iso_3166_1 . . . 1698
1.157.3 iso_3166_2 . . . 1700
1.157.4 iso_3166_protocol . . . 1701
1.158 iso_4217 . . . 1703
1.158.1 iso_4217 . . . 1703
1.158.2 iso_4217_protocol . . . 1704
1.159 iso_639 . . . 1706
1.159.1 iso_639 . . . 1706
1.159.2 iso_639_1 . . . 1707
1.159.3 iso_639_2 . . . 1709
1.159.4 iso_639_3 . . . 1710
1.159.5 iso_639_5 . . . 1711
1.159.6 iso_639_protocol . . . 1712
1.160 iso_9362 . . . 1715
1.160.1 iso_9362 . . . 1715
1.160.2 iso_9362_protocol . . . 1716
1.161 isolation_forest_anomaly_detector . . . 1718
1.161.1 isolation_forest_anomaly_detector . . . 1718
1.162 issue_creator . . . 1720
1.162.1 issue_creator . . . 1720
1.163 jaccard_recommender . . . 1729
1.163.1 jaccard_recommender . . . 1729

1.164 java . . . 1740
1.164.1 java . . . 1740
1.164.2 java(Reference) . . . 1741
1.164.3 java(Reference,ReturnValue) . . . 1742
1.164.4 java_access_protocol . . . 1743
1.164.5 java_hook . . . 1746
1.164.6 java_utils_protocol . . . 1747
1.165 json . . . 1755
1.165.1 json . . . 1756
1.165.2 json(StringRepresentation) . . . 1757
1.165.3 json(ObjectRepresentation,PairRepresentation,StringRepresentation) . . . 1758
1.165.4 json_protocol . . . 1759
1.166 json_graph . . . 1761
1.166.1 json_graph . . . 1761
1.166.2 json_graph_data_protocol . . . 1762
1.166.3 json_graph_protocol . . . 1764
1.167 json_ld . . . 1766
1.167.1 json_ld . . . 1766
1.167.2 json_ld(StringRepresentation) . . . 1767
1.167.3 json_ld(ObjectRepresentation,PairRepresentation,StringRepresentation) 1768
1.167.4 json_ld_protocol . . . 1769
1.168 json_lines . . . 1773
1.168.1 json_lines . . . 1773
1.168.2 json_lines(StringRepresentation) . . . 1774
1.168.3 json_lines(ObjectRepresentation,PairRepresentation,
StringRepresentation) . . . 1775
1.168.4 json_lines_protocol . . . 1776
1.169 json_patch . . . 1778
1.169.1 json_patch . . . 1778
1.169.2 json_patch_protocol . . . 1779
1.170 json_path . . . 1781
1.170.1 json_path . . . 1781
1.170.2 json_path(StringRepresentation) . . . 1782
1.170.3 json_path_protocol . . . 1783
1.171 json_pointer . . . 1787
1.171.1 json_pointer . . . 1787
1.171.2 json_pointer(StringRepresentation) . . . 1788
1.171.3 json_pointer_protocol . . . 1789
1.172 json_rpc . . . 1794
1.172.1 json_rpc . . . 1794
1.173 json_schema . . . 1806
1.173.1 json_schema . . . 1806
1.173.2 json_schema(StringRepresentation) . . . 1807
1.173.3 json_schema(ObjectRepresentation,PairRepresentation,
StringRepresentation) . . . 1808
1.173.4 json_schema_protocol . . . 1809
1.174 jwt . . . 1811
1.174.1 jwt . . . 1812
1.174.2 jwt_claims . . . 1822
1.174.3 jwt_compact . . . 1824
1.174.4 jwt_der . . . 1829
1.174.5 jwt_helpers . . . 1831
1.174.6 jwt_jwa . . . 1835
1.174.7 jwt_jwk . . . 1838
1.174.8 jwt_jwks . . . 1840
1.174.9 jwt_jws . . . 1844

1.174.10 jwt_openssl . . . 1847
1.175 kcenters_clusterer . . . 1848
1.175.1 kcenters_clusterer . . . 1848
1.176 kemeny_young_ranker . . . 1849
1.176.1 kemeny_young_ranker . . . 1849
1.177 kernel_pca_projection . . . 1852
1.177.1 kernel_pca_projection . . . 1852
1.178 kernel_svm_classifier . . . 1853
1.178.1 kernel_svm_classifier . . . 1853
1.179 kmeans_clusterer . . . 1854
1.179.1 kmeans_clusterer . . . 1854
1.180 kmedians_clusterer . . . 1856
1.180.1 kmedians_clusterer . . . 1856
1.181 kmedoids_clusterer . . . 1857
1.181.1 kmedoids_clusterer . . . 1857
1.182 kmodes_clusterer . . . 1858
1.182.1 kmodes_clusterer . . . 1858
1.183 knn_classifier . . . 1859
1.183.1 knn_classifier . . . 1860
1.184 knn_distance_anomaly_detector . . . 1861
1.184.1 knn_distance_anomaly_detector . . . 1861
1.185 knn_forecasting . . . 1863
1.185.1 knn_forecasting . . . 1863
1.186 knn_item_recommender . . . 1864
1.186.1 knn_item_recommender . . . 1865
1.187 knn_regression . . . 1866
1.187.1 knn_regression . . . 1866
1.188 knn_user_recommender . . . 1867
1.188.1 knn_user_recommender . . . 1867
1.189 kprototypes_clusterer . . . 1868
1.189.1 kprototypes_clusterer . . . 1868
1.190 ksuid . . . 1870
1.190.1 ksuid . . . 1870
1.190.2 ksuid(Representation,Alphabet) . . . 1871
1.190.3 ksuid_protocol . . . 1872
1.191 language_detection . . . 1873
1.191.1 af_language_profile . . . 1873
1.191.2 ar_language_profile . . . 1874
1.191.3 bg_language_profile . . . 1875
1.191.4 bn_language_profile . . . 1876
1.191.5 br_language_profile . . . 1877
1.191.6 ca_language_profile . . . 1878
1.191.7 cs_language_profile . . . 1879
1.191.8 da_language_profile . . . 1880
1.191.9 de_language_profile . . . 1881
1.191.10 el_language_profile . . . 1882
1.191.11 en_language_profile . . . 1883
1.191.12 eo_language_profile . . . 1884
1.191.13 es_language_profile . . . 1885
1.191.14 et_language_profile . . . 1886
1.191.15 eu_language_profile . . . 1887
1.191.16 fa_language_profile . . . 1888
1.191.17 fi_language_profile . . . 1889
1.191.18 fr_language_profile . . . 1890
1.191.19 ga_language_profile . . . 1891
1.191.20 gl_language_profile . . . 1892

1.191.21 ha_language_profile . . . 1893
1.191.22 he_language_profile . . . 1894
1.191.23 hi_language_profile . . . 1895
1.191.24 hr_language_profile . . . 1896
1.191.25 hu_language_profile . . . 1897
1.191.26 hy_language_profile . . . 1898
1.191.27 hybrid_language_detector . . . 1899
1.191.28 id_language_profile . . . 1901
1.191.29 it_language_profile . . . 1902
1.191.30 ja_language_profile . . . 1903
1.191.31 ko_language_profile . . . 1904
1.191.32 ku_language_profile . . . 1905
1.191.33 la_language_profile . . . 1906
1.191.34 language_detection_scoring . . . 1907
1.191.35 language_detection_scripts . . . 1909
1.191.36 language_detection_strategy_protocol . . . 1912
1.191.37 language_detection_tokenizer . . . 1913
1.191.38 language_detector(Representation,Strategy) . . . 1914
1.191.39 language_detector_protocol . . . 1915
1.191.40 language_profile_common . . . 1920
1.191.41 language_profile_expansion . . . 1922
1.191.42 language_profile_protocol . . . 1923
1.191.43 language_profiles . . . 1925
1.191.44 lt_language_profile . . . 1927
1.191.45 lv_language_profile . . . 1928
1.191.46 mr_language_profile . . . 1929
1.191.47 ms_language_profile . . . 1930
1.191.48 ngram_language_detector . . . 1931
1.191.49 nl_language_profile . . . 1932
1.191.50 no_language_profile . . . 1933
1.191.51 pl_language_profile . . . 1934
1.191.52 pt_language_profile . . . 1935
1.191.53 ro_language_profile . . . 1936
1.191.54 ru_language_profile . . . 1937
1.191.55 script_language_detector . . . 1938
1.191.56 sk_language_profile . . . 1939
1.191.57 sl_language_profile . . . 1940
1.191.58 so_language_profile . . . 1941
1.191.59 st_language_profile . . . 1942
1.191.60 stopword_language_detector . . . 1943
1.191.61 sv_language_profile . . . 1944
1.191.62 sw_language_profile . . . 1945
1.191.63 th_language_profile . . . 1946
1.191.64 tl_language_profile . . . 1947
1.191.65 tr_language_profile . . . 1948
1.191.66 uk_language_profile . . . 1949
1.191.67 ur_language_profile . . . 1950
1.191.68 vi_language_profile . . . 1951
1.191.69 yo_language_profile . . . 1952
1.191.70 zh_language_profile . . . 1953
1.191.71 zu_language_profile . . . 1954
1.192 lasso_feature_selector . . . 1955
1.192.1 lasso_feature_selector . . . 1955
1.192.2 regression_dataset_adapter(Dataset) . . . 1962
1.192.3 regression_examples_adapter(Declarations,Examples) . . . 1964
1.193 lasso_regression . . . 1965

1.193.1 lasso_regression . . . 1965
1.194 lda_classifier . . . 1966
1.194.1 lda_classifier . . . 1966
1.195 lda_projection . . . 1968
1.195.1 lda_projection . . . 1968
1.196 lemmatization . . . 1969
1.196.1 english_lemmatizer . . . 1969
1.196.2 lemmatizer(Representation,Language) . . . 1970
1.196.3 lemmatizer_language_protocol . . . 1971
1.196.4 lemmatizer_protocol . . . 1973
1.197 lgtdoc . . . 1978
1.197.1 lgtdoc . . . 1978
1.197.2 lgtdoc_messages . . . 1982
1.197.3 lgtdocp . . . 1983
1.198 lgtunit . . . 1992
1.198.1 automation_report . . . 1992
1.198.2 cobertura_report . . . 1993
1.198.3 coverage_report . . . 1996
1.198.4 ctrf_output . . . 1998
1.198.5 ctrf_report . . . 1999
1.198.6 lcov_report . . . 2000
1.198.7 lgtunit . . . 2002
1.198.8 lgtunit_messages . . . 2044
1.198.9 minimal_output . . . 2045
1.198.10 sonar_report . . . 2046
1.198.11 subunit_v1_output . . . 2047
1.198.12 subunit_v1_report . . . 2048
1.198.13 subunit_v2_output . . . 2049
1.198.14 subunit_v2_report . . . 2051
1.198.15 tap_output . . . 2052
1.198.16 tap_report . . . 2054
1.198.17 xunit_net_v2_output . . . 2055
1.198.18 xunit_net_v2_report . . . 2057
1.198.19 xunit_output . . . 2058
1.198.20 xunit_report . . . 2059
1.199 library . . . 2061
1.199.1 cloning . . . 2061
1.199.2 counters . . . 2062
1.199.3 streamvars . . . 2065
1.200 linda . . . 2067
1.200.1 linda . . . 2067
1.200.2 linda_client . . . 2069
1.200.3 linda_server . . . 2079
1.201 linear_algebra . . . 2087
1.201.1 linear_algebra . . . 2087
1.201.2 linear_algebra_protocol . . . 2088
1.202 linear_programming . . . 2121
1.202.1 linear_programming_common . . . 2121
1.202.2 linear_programming_protocol . . . 2124
1.202.3 linear_programming_types . . . 2133
1.202.4 milp_branch_and_bound . . . 2134
1.202.5 simplex . . . 2135
1.203 linear_regression . . . 2136
1.203.1 linear_regression . . . 2136
1.204 linear_svm_classifier . . . 2137
1.204.1 linear_svm_classifier . . . 2138

1.205 linter_reporter . . . 2139
1.205.1 linter_reporter . . . 2139
1.206 listing . . . 2144
1.206.1 listing . . . 2144
1.207 local_optimization . . . 2146
1.207.1 barzilai_borwein(Problem) . . . 2146
1.207.2 bfgs(Problem) . . . 2148
1.207.3 conjugate_gradient(Problem) . . . 2149
1.207.4 gradient_descent(Problem) . . . 2150
1.207.5 lbfgs(Problem) . . . 2152
1.207.6 lbfgs_b(Problem) . . . 2153
1.207.7 local_optimization_problem_protocol . . . 2155
1.207.8 local_optimization_solver(Problem) . . . 2158
1.207.9 nelder_mead(Problem) . . . 2164
1.207.10 trust_region_newton_cg(Problem) . . . 2165
1.208 lof_anomaly_detector . . . 2166
1.208.1 lof_anomaly_detector . . . 2167
1.209 logging . . . 2169
1.209.1 logger . . . 2169
1.209.2 logging . . . 2171
1.209.3 loggingp . . . 2172
1.210 logistic_regression_classifier . . . 2176
1.210.1 logistic_regression_classifier . . . 2176
1.211 loops . . . 2177
1.211.1 loop . . . 2177
1.211.2 loopp . . . 2178
1.212 lru_caches . . . 2183
1.212.1 lru_cache . . . 2183
1.212.2 lru_cache_protocol . . . 2184
1.213 massey_ranker . . . 2190
1.213.1 massey_ranker . . . 2190
1.214 mcp_server . . . 2192
1.214.1 mcp_cache_protocol . . . 2192
1.214.2 mcp_completion_protocol . . . 2193
1.214.3 mcp_multiround_protocol . . . 2195
1.214.4 mcp_prompt_protocol . . . 2197
1.214.5 mcp_resource_protocol . . . 2199
1.214.6 mcp_server . . . 2202
1.214.7 mcp_server_2025_06_18_spec . . . 2205
1.214.8 mcp_server_2025_11_25_spec . . . 2213
1.214.9 mcp_server_2026_07_28_spec . . . 2214
1.214.10 mcp_server_application . . . 2218
1.214.11 mcp_server_spec_protocol . . . 2225
1.214.12 mcp_server_stdio_transport . . . 2228
1.214.13 mcp_server_streamable_http_transport . . . 2229
1.214.14 mcp_server_transport_protocol . . . 2236
1.214.15 mcp_streamable_http_handler . . . 2238
1.214.16 mcp_tool_protocol . . . 2239
1.214.17 mcp_ui_protocol . . . 2243
1.215 memcached . . . 2245
1.215.1 memcached . . . 2245
1.216 message_pack . . . 2257
1.216.1 message_pack(StringRepresentation) . . . 2257
1.217 meta . . . 2259
1.217.1 meta . . . 2259
1.217.2 metap . . . 2260

1.218 meta_compiler . . . 2269
1.218.1 meta_compiler . . . 2269
1.219 metagol . . . 2271
1.219.1 metagol . . . 2271
1.219.2 metagol_example_protocol . . . 2279
1.220 mime_types . . . 2281
1.220.1 mime_types . . . 2281
1.221 minizinc . . . 2288
1.221.1 minizinc . . . 2288
1.221.2 minizinc_expansion . . . 2289
1.221.3 minizinc_output . . . 2290
1.221.4 minizinc_problem_protocol . . . 2292
1.221.5 minizinc_process . . . 2293
1.221.6 minizinc_program . . . 2296
1.221.7 minizinc_protocol . . . 2298
1.222 mlp_classifier . . . 2305
1.222.1 mlp_classifier . . . 2305
1.223 modified_z_score_anomaly_detector . . . 2306
1.223.1 modified_z_score_anomaly_detector . . . 2307
1.224 mqtt . . . 2308
1.224.1 mqtt . . . 2308
1.225 mrmr_feature_selector . . . 2328
1.225.1 mrmr_feature_selector . . . 2328
1.226 multisets . . . 2329
1.226.1 multisets . . . 2329
1.226.2 multisets_protocol . . . 2330
1.227 multivariate_distributions . . . 2339
1.227.1 multivariate_distributions(Random) . . . 2339
1.227.2 multivariate_distributions_protocol . . . 2341
1.228 mutation_testing . . . 2371
1.228.1 arithmetic_operator_replacement(Entity,Predicate,ClauseIndex, Occurrence,PrintMutation) . . . 2371
1.228.2 body_goal_negation(Entity,Predicate,ClauseIndex,Occurrence, PrintMutation) . . . 2372
1.228.3 clause_mutator_protocol . . . 2373
1.228.4 clauses_reordering(Entity,Predicate,ClauseIndex,Occurrence, PrintMutation) . . . 2374
1.228.5 directive_mutator_protocol . . . 2376
1.228.6 fail_insertion(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation) 2377
1.228.7 head_arguments_mutation(Entity,Predicate,ClauseIndex,Occurrence, PrintMutation) . . . 2378
1.228.8 head_arguments_reordering(Entity,Predicate,ClauseIndex,Occurrence, PrintMutation) . . . 2379
1.228.9 mutation_testing . . . 2380
1.228.10 mutation_testing_messages . . . 2391
1.228.11 mutator_common . . . 2392
1.228.12 mutator_protocol . . . 2399
1.228.13 predicate_directive_suppression(Entity,Predicate,DirectiveIndex, Occurrence,PrintMutation) . . . 2401
1.228.14 relational_operator_inversion(Entity,Predicate,ClauseIndex, Occurrence,PrintMutation) . . . 2402
1.228.15 relational_operator_replacement(Entity,Predicate,ClauseIndex, Occurrence,PrintMutation) . . . 2403
1.228.16 scope_directive_replacement(Entity,Predicate,DirectiveIndex, Occurrence,PrintMutation) . . . 2404
1.228.17 subprocess_coverage_hook . . . 2406

1.228.18 subprocess_mutation_hook . . . 2407
1.228.19 truth_literal_flip(Entity,Predicate,ClauseIndex,Occurrence, PrintMutation) . . . 2409
1.228.20 uses_directive_resource_deletion(Entity,Predicate,DirectiveIndex, Occurrence,PrintMutation) . . . 2410
1.229 mutations . . . 2412
1.229.1 default_atom_mutations . . . 2412
1.229.2 default_compound_mutations . . . 2413
1.229.3 default_float_mutations . . . 2414
1.229.4 default_integer_mutations . . . 2415
1.229.5 default_list_mutations . . . 2416
1.229.6 mutations . . . 2417
1.229.7 mutations_store . . . 2418
1.230 mutual_information_feature_selector . . . 2421
1.230.1 mutual_information_feature_selector . . . 2421
1.231 n_grams . . . 2422
1.231.1 n_grams(Representation) . . . 2422
1.231.2 n_grams_protocol . . . 2423
1.232 naive_bayes_classifier . . . 2430
1.232.1 naive_bayes_classifier . . . 2430
1.233 nanoid . . . 2431
1.233.1 nanoid . . . 2431
1.233.2 nanoid(Representation,Size,Alphabet) . . . 2432
1.233.3 nanoid_protocol . . . 2433
1.234 nearest_centroid_classifier . . . 2434
1.234.1 nearest_centroid_classifier . . . 2435
1.235 nested_dictionaries . . . 2437
1.235.1 navltree . . . 2437
1.235.2 nbintree . . . 2438
1.235.3 nested_dictionary_protocol . . . 2439
1.235.4 nrbtree . . . 2443
1.236 nmea . . . 2444
1.236.1 nmea . . . 2444
1.236.2 nmea_protocol . . . 2445
1.237 nmf_projection . . . 2449
1.237.1 nmf_projection . . . 2449
1.238 numerical_analysis . . . 2450
1.238.1 adaptive_simpson_quadrature(Function) . . . 2450
1.238.2 barycentric_interpolator . . . 2451
1.238.3 bisection_root_finder(Function) . . . 2453
1.238.4 brent_root_finder(Function) . . . 2454
1.238.5 cubic_spline_interpolator . . . 2455
1.238.6 euler_ode_solver(System) . . . 2457
1.238.7 gauss_legendre_quadrature(Function) . . . 2458
1.238.8 interpolator . . . 2459
1.238.9 interpolator_protocol . . . 2461
1.238.10 newton_root_finder(Function) . . . 2464
1.238.11 ode_solver(System) . . . 2465
1.238.12 ode_solver_protocol . . . 2467
1.238.13 ode_system_protocol . . . 2470
1.238.14 piecewise_linear_interpolator . . . 2471
1.238.15 quadrature(Function) . . . 2472
1.238.16 quadrature_protocol . . . 2474
1.238.17 rk45_ode_solver(System) . . . 2477
1.238.18 rk4_ode_solver(System) . . . 2478
1.238.19 root_finder(Function) . . . 2479

1.238.20 root_finder_protocol . . . 2481
1.238.21 secant_root_finder(Function) . . . 2484
1.238.22 univariate_function_protocol . . . 2485
1.239 omt . . . 2486
1.239.1 omt . . . 2487
1.239.2 omt_expansion . . . 2488
1.239.3 omt_problem_protocol . . . 2489
1.239.4 omt_protocol . . . 2490
1.240 one_class_svm_anomaly_detector . . . 2496
1.240.1 one_class_svm_anomaly_detector . . . 2496
1.241 open_ai . . . 2497
1.241.1 open_ai . . . 2497
1.241.2 open_ai_api . . . 2499
1.241.3 open_ai_backend_protocol . . . 2500
1.241.4 open_ai_catalog . . . 2501
1.241.5 open_ai_client . . . 2503
1.241.6 open_ai_event_stream . . . 2505
1.241.7 open_ai_server(Backend) . . . 2507
1.242 open_api . . . 2511
1.242.1 open_api . . . 2511
1.242.2 open_api_provider_protocol . . . 2522
1.243 open_id . . . 2525
1.243.1 open_id . . . 2525
1.243.2 open_id_client . . . 2535
1.243.3 open_id_der . . . 2540
1.243.4 open_id_discovery . . . 2542
1.243.5 open_id_helpers . . . 2545
1.243.6 open_id_jwks . . . 2553
1.243.7 open_id_jwks_cache . . . 2555
1.243.8 open_id_jwt . . . 2559
1.243.9 open_id_logout . . . 2562
1.243.10 open_id_openssl . . . 2564
1.243.11 open_id_pkce . . . 2565
1.243.12 open_id_response . . . 2568
1.244 optics_clusterer . . . 2570
1.244.1 optics_clusterer . . . 2570
1.245 optionals . . . 2572
1.245.1 maybe . . . 2572
1.245.2 optional . . . 2574
1.245.3 optional(Optional) . . . 2578
1.246 options . . . 2586
1.246.1 options . . . 2586
1.246.2 options_protocol . . . 2587
1.247 os . . . 2592
1.247.1 os . . . 2592
1.247.2 os_types . . . 2593
1.247.3 osp . . . 2595
1.248 otp . . . 2611
1.248.1 otp . . . 2611
1.248.2 otp_protocol . . . 2612
1.249 packs . . . 2616
1.249.1 pack_protocol . . . 2616
1.249.2 packs . . . 2619
1.249.3 packs_common . . . 2650
1.249.4 packs_messages . . . 2660
1.249.5 packs_specs_hook . . . 2661

1.249.6 registries . . . 2662
1.249.7 registry_loader_hook . . . 2672
1.249.8 registry_protocol . . . 2673
1.250 particle_swarm_optimization . . . 2676
1.250.1 particle_swarm_optimization(Problem) . . . 2676
1.250.2 particle_swarm_optimization(Problem,Random) . . . 2677
1.250.3 particle_swarm_optimization_protocol . . . 2680
1.251 partitions . . . 2683
1.251.1 partitions . . . 2683
1.251.2 partitions_protocol . . . 2684
1.252 paseto . . . 2699
1.252.1 paseto . . . 2699
1.252.2 paseto_claims . . . 2708
1.252.3 paseto_claims_helpers . . . 2711
1.252.4 paseto_helpers . . . 2714
1.252.5 paseto_keys . . . 2716
1.252.6 paseto_protocol . . . 2718
1.252.7 paseto_v4 . . . 2727
1.253 pattern_mining_protocols . . . 2729
1.253.1 pattern_miner_common . . . 2729
1.253.2 pattern_miner_protocol . . . 2732
1.254 pca_projection . . . 2737
1.254.1 pca_projection . . . 2737
1.255 pddl_parser . . . 2738
1.255.1 pddl . . . 2738
1.255.2 read_file . . . 2740
1.256 permutations . . . 2742
1.256.1 permutations . . . 2742
1.256.2 permutations_protocol . . . 2743
1.257 plackett_luce_last_ranker . . . 2753
1.257.1 plackett_luce_last_ranker . . . 2753
1.258 plackett_luce_ranker . . . 2754
1.258.1 plackett_luce_ranker . . . 2755
1.259 pls_projection . . . 2756
1.259.1 pls_projection . . . 2756
1.260 ports_profiler . . . 2757
1.260.1 ports_profiler . . . 2758
1.261 prefix_span_pattern_miner . . . 2763
1.261.1 prefix_span_pattern_miner . . . 2763
1.262 probabilistic_pca_projection . . . 2764
1.262.1 probabilistic_pca_projection . . . 2764
1.263 problog . . . 2765
1.263.1 problog . . . 2766
1.263.2 problog_expansion . . . 2767
1.263.3 problog_output . . . 2768
1.263.4 problog_problem_protocol . . . 2770
1.263.5 problog_process . . . 2771
1.263.6 problog_program . . . 2773
1.263.7 problog_protocol . . . 2776
1.264 process . . . 2781
1.264.1 process . . . 2781
1.265 protobuf . . . 2784
1.265.1 protobuf . . . 2784
1.266 qda_classifier . . . 2787
1.266.1 qda_classifier . . . 2787
1.267 queues . . . 2788

1.267.1 queue . . . 2788
1.267.2 queuep . . . 2789
1.268 random . . . 2795
1.268.1 backend_random . . . 2795
1.268.2 fast_random . . . 2796
1.268.3 fast_random(Algorithm) . . . 2797
1.268.4 pseudo_random_protocol . . . 2800
1.268.5 random . . . 2802
1.268.6 random(Algorithm) . . . 2803
1.268.7 random_protocol . . . 2806
1.268.8 sampling_protocol . . . 2814
1.269 random_forest_classifier . . . 2826
1.269.1 random_forest_classifier . . . 2826
1.270 random_forest_regression . . . 2827
1.270.1 random_forest_regression . . . 2827
1.271 random_projection . . . 2828
1.271.1 random_projection . . . 2828
1.272 rank_centrality . . . 2830
1.272.1 rank_centrality . . . 2830
1.273 ranked_pairs . . . 2831
1.273.1 ranked_pairs . . . 2831
1.274 ranking_protocols . . . 2833
1.274.1 condorcet_victory_common . . . 2833
1.274.2 glicko2_common . . . 2836
1.274.3 grouped_strength_ranker_common . . . 2838
1.274.4 multiplayer_ranking_dataset_protocol . . . 2842
1.274.5 pairwise_measurement_dataset_protocol . . . 2844
1.274.6 pairwise_ranking_dataset_protocol . . . 2846
1.274.7 pairwise_strength_ranker_common . . . 2847
1.274.8 plackett_luce_common . . . 2852
1.274.9 ranker_common . . . 2855
1.274.10 ranker_protocol . . . 2858
1.274.11 ranking_dataset_common . . . 2863
1.274.12 ranking_dataset_protocol . . . 2880
1.274.13 score_ranker_common . . . 2882
1.274.14 score_ranker_model_common . . . 2883
1.274.15 temporal_pairwise_ranking_dataset_protocol . . . 2888
1.275 reader . . . 2890
1.275.1 reader . . . 2890
1.276 recommender_protocols . . . 2898
1.276.1 cosine_similarity . . . 2899
1.276.2 item_content_dataset_protocol . . . 2900
1.276.3 item_content_dataset_validation . . . 2902
1.276.4 jaccard_similarity . . . 2905
1.276.5 msd_similarity . . . 2906
1.276.6 pearson_similarity . . . 2907
1.276.7 rating_dataset_protocol . . . 2908
1.276.8 recommender_common . . . 2910
1.276.9 recommender_protocol . . . 2922
1.276.10 similarity_metric_common . . . 2931
1.276.11 similarity_metric_protocol . . . 2936
1.276.12 spearman_similarity . . . 2937
1.277 recorded_database . . . 2938
1.277.1 recorded_database . . . 2938
1.277.2 recorded_database_core . . . 2939
1.278 redis . . . 2944

1.278.1 redis . . . 2944
1.279 regression_protocols . . . 2963
1.279.1 regression_dataset_protocol . . . 2964
1.279.2 regressor_common . . . 2965
1.279.3 regressor_protocol . . . 2991
1.280 regression_tree . . . 2996
1.280.1 regression_tree . . . 2996
1.281 regularized_bradley_terry_ranker . . . 2997
1.281.1 regularized_bradley_terry_ranker . . . 2997
1.282 relief_feature_selector . . . 2999
1.282.1 relief_feature_selector . . . 2999
1.283 relieff_feature_selector . . . 3000
1.283.1 relieff_feature_selector . . . 3000
1.284 rest . . . 3001
1.284.1 rest . . . 3001
1.285 ridge_regression . . . 3012
1.285.1 ridge_regression . . . 3012
1.286 ripple_down_rules . . . 3013
1.286.1 generalized_ripple_down_rules . . . 3013
1.286.2 multi_classification_ripple_down_rules . . . 3016
1.286.3 ripple_down_rules_common_protocol . . . 3017
1.286.4 ripple_down_rules_protocol . . . 3021
1.286.5 single_classification_ripple_down_rules . . . 3022
1.287 rrelieff_feature_selector . . . 3023
1.287.1 rrelieff_feature_selector . . . 3023
1.288 s3 . . . 3024
1.288.1 s3_client . . . 3025
1.288.2 s3_client(HTTPTransport) . . . 3026
1.288.3 s3_endpoints . . . 3027
1.288.4 s3_headers . . . 3030
1.288.5 s3_protocol . . . 3032
1.288.6 s3_session(HTTPTransport) . . . 3045
1.288.7 s3_signing . . . 3058
1.288.8 s3_xml . . . 3063
1.289 sarif . . . 3066
1.289.1 sarif . . . 3066
1.290 sat . . . 3068
1.290.1 sat . . . 3068
1.290.2 sat_expansion . . . 3070
1.290.3 sat_output . . . 3071
1.290.4 sat_problem_protocol . . . 3073
1.290.5 sat_process . . . 3074
1.290.6 sat_program . . . 3077
1.290.7 sat_protocol . . . 3079
1.291 sbom . . . 3088
1.291.1 sbom . . . 3088
1.292 schulze_ranker . . . 3091
1.292.1 schulze_ranker . . . 3091
1.293 sequential_pattern_mining_protocols . . . 3092
1.293.1 sequence_dataset_protocol . . . 3093
1.293.2 sequential_pattern_mining_common . . . 3094
1.294 sets . . . 3098
1.294.1 set . . . 3098
1.294.2 set(Type) . . . 3099
1.294.3 setp . . . 3101
1.294.4 treap_set . . . 3110

1.295 sgd_classifier . . . 3111
1.295.1 sgd_classifier . . . 3111
1.296 simple_temporal_networks . . . 3112
1.296.1 stn . . . 3113
1.296.2 stn_protocol . . . 3114
1.297 simulated_annealing . . . 3127
1.297.1 simulated_annealing(Problem) . . . 3128
1.297.2 simulated_annealing(Problem,Random) . . . 3129
1.297.3 simulated_annealing_protocol . . . 3132
1.298 slope_one_recommender . . . 3136
1.298.1 slope_one_recommender . . . 3136
1.299 smt . . . 3137
1.299.1 smt . . . 3137
1.299.2 smt_expansion . . . 3138
1.299.3 smt_output . . . 3139
1.299.4 smt_problem_protocol . . . 3141
1.299.5 smt_process . . . 3143
1.299.6 smt_program . . . 3145
1.299.7 smt_protocol . . . 3148
1.300 smtp . . . 3157
1.300.1 smtp . . . 3157
1.300.2 smtp_process_transport . . . 3158
1.300.3 smtp_protocol . . . 3161
1.301 snowflakeid . . . 3166
1.301.1 snowflakeid . . . 3166
1.301.2 snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds, TimestampBits,NodeBits,SequenceBits,Node) . . . 3167
1.301.3 snowflakeid_instagram . . . 3169
1.301.4 snowflakeid_instagram(Representation) . . . 3170
1.301.5 snowflakeid_protocol . . . 3171
1.301.6 snowflakeid_sonyflake . . . 3172
1.301.7 snowflakeid_sonyflake(Representation) . . . 3173
1.301.8 snowflakeid_twitter . . . 3174
1.301.9 snowflakeid_twitter(Representation) . . . 3175
1.302 sockets . . . 3176
1.302.1 socket . . . 3176
1.303 souffle . . . 3182
1.303.1 souffle . . . 3182
1.303.2 souffle_expansion . . . 3183
1.303.3 souffle_output . . . 3184
1.303.4 souffle_problem_protocol . . . 3186
1.303.5 souffle_process . . . 3187
1.303.6 souffle_program . . . 3190
1.303.7 souffle_protocol . . . 3193
1.304 spade_pattern_miner . . . 3200
1.304.1 spade_pattern_miner . . . 3200
1.305 spectral_clusterer . . . 3201
1.305.1 spectral_clusterer . . . 3201
1.306 sqids . . . 3203
1.306.1 sqids . . . 3203
1.306.2 sqids(Representation) . . . 3204
1.307 statistics . . . 3208
1.307.1 population . . . 3208
1.307.2 sample . . . 3209
1.307.3 statistics . . . 3210
1.307.4 statisticsp . . . 3216

1.308 stemming . . . 3230
1.308.1 lovins_stemmer(Representation) . . . 3230
1.308.2 porter_stemmer(Representation) . . . 3231
1.308.3 stemmer_protocol . . . 3233
1.309 stomp . . . 3234
1.309.1 stomp . . . 3234
1.310 stop_words . . . 3242
1.310.1 stop_words(Representation,Language) . . . 3243
1.310.2 stop_words_language_protocol . . . 3244
1.310.3 stop_words_protocol . . . 3245
1.310.4 stopwords_af . . . 3247
1.310.5 stopwords_ar . . . 3248
1.310.6 stopwords_bg . . . 3250
1.310.7 stopwords_bn . . . 3251
1.310.8 stopwords_br . . . 3252
1.310.9 stopwords_ca . . . 3253
1.310.10 stopwords_cs . . . 3254
1.310.11 stopwords_da . . . 3255
1.310.12 stopwords_de . . . 3257
1.310.13 stopwords_el . . . 3258
1.310.14 stopwords_en . . . 3259
1.310.15 stopwords_eo . . . 3260
1.310.16 stopwords_es . . . 3261
1.310.17 stopwords_et . . . 3262
1.310.18 stopwords_eu . . . 3264
1.310.19 stopwords_fa . . . 3265
1.310.20 stopwords_fi . . . 3266
1.310.21 stopwords_fr . . . 3267
1.310.22 stopwords_ga . . . 3268
1.310.23 stopwords_gl . . . 3269
1.310.24 stopwords_ha . . . 3271
1.310.25 stopwords_he . . . 3272
1.310.26 stopwords_hi . . . 3273
1.310.27 stopwords_hr . . . 3274
1.310.28 stopwords_hu . . . 3275
1.310.29 stopwords_hy . . . 3276
1.310.30 stopwords_id . . . 3278
1.310.31 stopwords_it . . . 3279
1.310.32 stopwords_ja . . . 3280
1.310.33 stopwords_ko . . . 3281
1.310.34 stopwords_ku . . . 3282
1.310.35 stopwords_la . . . 3283
1.310.36 stopwords_lt . . . 3285
1.310.37 stopwords_lv . . . 3286
1.310.38 stopwords_mr . . . 3287
1.310.39 stopwords_ms . . . 3288
1.310.40 stopwords_nl . . . 3289
1.310.41 stopwords_no . . . 3290
1.310.42 stopwords_pl . . . 3292
1.310.43 stopwords_pt . . . 3293
1.310.44 stopwords_ro . . . 3294
1.310.45 stopwords_ru . . . 3295
1.310.46 stopwords_sk . . . 3296
1.310.47 stopwords_sl . . . 3297
1.310.48 stopwords_so . . . 3299
1.310.49 stopwords_st . . . 3300

1.310.50 stopwords_sv . . . 3301
1.310.51 stopwords_sw . . . 3302
1.310.52 stopwords_th . . . 3303
1.310.53 stopwords_tl . . . 3304
1.310.54 stopwords_tr . . . 3306
1.310.55 stopwords_uk . . . 3307
1.310.56 stopwords_ur . . . 3308
1.310.57 stopwords_vi . . . 3309
1.310.58 stopwords_yo . . . 3310
1.310.59 stopwords_zh . . . 3311
1.310.60 stopwords_zu . . . 3313
1.311 string_distance . . . 3314
1.311.1 string_distance(Representation) . . . 3314
1.312 strings . . . 3323
1.312.1 string(Representation) . . . 3323
1.313 subsequences . . . 3330
1.313.1 subsequences . . . 3331
1.313.2 subsequences_protocol . . . 3332
1.314 svr_regression . . . 3350
1.314.1 svr_regression . . . 3351
1.315 tabu_search . . . 3352
1.315.1 tabu_search(Problem) . . . 3352
1.315.2 tabu_search(Problem,Random) . . . 3353
1.315.3 tabu_search_problem_protocol . . . 3358
1.316 term_io . . . 3363
1.316.1 term_io . . . 3363
1.316.2 term_io_protocol . . . 3365
1.317 text_normalization . . . 3377
1.317.1 default_text_normalization . . . 3377
1.317.2 spelling_normalizer_protocol . . . 3378
1.317.3 text_case_folding . . . 3380
1.317.4 text_diacritics . . . 3381
1.317.5 text_entities . . . 3383
1.317.6 text_normalization_profile_protocol . . . 3384
1.317.7 text_normalizer(Representation,Profile) . . . 3386
1.317.8 text_normalizer_protocol . . . 3388
1.317.9 text_unicode . . . 3400
1.317.10 text_whitespace . . . 3401
1.317.11 xml_whatwg_entities . . . 3403
1.318 text_vectorization . . . 3404
1.318.1 text_vectorizer . . . 3404
1.318.2 text_vectorizer_protocol . . . 3405
1.319 tfidf_recommender . . . 3413
1.319.1 tfidf_recommender . . . 3413
1.320 theta_forecasting . . . 3425
1.320.1 theta_forecasting . . . 3425
1.321 thurstone_mosteller_ranker . . . 3435
1.321.1 thurstone_mosteller_ranker . . . 3435
1.322 time_scales . . . 3436
1.322.1 time_scales . . . 3436
1.322.2 time_scales_data . . . 3438
1.322.3 time_scales_protocol . . . 3446
1.323 time_series_protocols . . . 3456
1.323.1 forecaster_common . . . 3457
1.323.2 forecaster_protocol . . . 3487
1.323.3 time_series_dataset_protocol . . . 3494

1.324 time_series_regression . . . 3495
1.324.1 time_series_regression . . . 3496
1.325 timeout . . . 3498
1.325.1 timeout . . . 3498
1.326 tle_orbits . . . 3500
1.326.1 tle_orbits . . . 3500
1.326.2 tle_orbits_protocol . . . 3501
1.327 tokenization . . . 3507
1.327.1 english_tokenizer . . . 3507
1.327.2 sentence_splitter_protocol . . . 3508
1.327.3 tokenizer(Representation,Language) . . . 3511
1.327.4 tokenizer_language_protocol . . . 3512
1.327.5 tokenizer_protocol . . . 3514
1.327.6 tokenizer_rules . . . 3518
1.328 toml . . . 3521
1.328.1 toml . . . 3521
1.328.2 toml(StringRepresentation) . . . 3522
1.328.3 toml(ObjectRepresentation,PairRepresentation,StringRepresentation) . . . 3523
1.328.4 toml_protocol . . . 3524
1.329 tool_diagnostics . . . 3526
1.329.1 tool_diagnostics_common . . . 3526
1.329.2 tool_diagnostics_protocol . . . 3527
1.330 toon . . . 3533
1.330.1 toon . . . 3533
1.330.2 toon(StringRepresentation) . . . 3534
1.330.3 toon(ObjectRepresentation,PairRepresentation,StringRepresentation) . . . 3535
1.330.4 toon_protocol . . . 3536
1.331 toychr . . . 3538
1.331.1 toychrdb . . . 3538
1.332 tries . . . 3545
1.332.1 trie(Representation) . . . 3545
1.333 trueskill_ranker . . . 3552
1.333.1 trueskill_ranker . . . 3553
1.334 truncated_svd_projection . . . 3554
1.334.1 truncated_svd_projection . . . 3554
1.335 tsne_projection . . . 3556
1.335.1 tsne_projection . . . 3556
1.336 tsv . . . 3557
1.336.1 tsv . . . 3557
1.336.2 tsv(Header) . . . 3558
1.336.3 tsv(Header,Comments) . . . 3559
1.336.4 tsv_protocol . . . 3560
1.337 tutor . . . 3568
1.337.1 tutor . . . 3568
1.337.2 tutor_explanations . . . 3569
1.338 typeid . . . 3571
1.338.1 typeid . . . 3571
1.338.2 typeid(Representation) . . . 3572
1.338.3 typeid_protocol . . . 3573
1.339 types . . . 3578
1.339.1 atom . . . 3578
1.339.2 atomic . . . 3580
1.339.3 callable . . . 3581
1.339.4 character . . . 3582
1.339.5 characterp . . . 3583
1.339.6 comparingp . . . 3591

1.339.7 compound . . . 3594
1.339.8 difflist . . . 3595
1.339.9 float . . . 3597
1.339.10 integer . . . 3599
1.339.11 list . . . 3602
1.339.12 list(Type) . . . 3603
1.339.13 listp . . . 3604
1.339.14 natural . . . 3625
1.339.15 number . . . 3630
1.339.16 numberlist . . . 3633
1.339.17 numberlistp . . . 3635
1.339.18 pairs . . . 3645
1.339.19 term . . . 3649
1.339.20 termp . . . 3650
1.339.21 type . . . 3656
1.339.22 varlist . . . 3661
1.339.23 varlistp . . . 3662
1.340 tzif . . . 3671
1.340.1 tzif . . . 3671
1.340.2 tzif_protocol . . . 3673
1.340.3 tzif_zone_ids . . . 3696
1.341 ulid . . . 3698
1.341.1 ulid . . . 3698
1.341.2 ulid(Representation) . . . 3699
1.341.3 ulid_protocol . . . 3700
1.341.4 ulid_types . . . 3704
1.342 umap_projection . . . 3705
1.342.1 umap_projection . . . 3705
1.343 union_find . . . 3707
1.343.1 union_find . . . 3707
1.343.2 union_find_protocol . . . 3708
1.344 univariate_distributions . . . 3712
1.344.1 univariate_distributions(Random) . . . 3712
1.344.2 univariate_distributions_protocol . . . 3713
1.345 uri_template . . . 3748
1.345.1 uri_template(Representation) . . . 3748
1.346 url . . . 3750
1.346.1 url(Representation) . . . 3750
1.347 uuid . . . 3755
1.347.1 uuid . . . 3755
1.347.2 uuid(Representation) . . . 3756
1.347.3 uuid_protocol . . . 3757
1.348 validations . . . 3762
1.348.1 validated . . . 3762
1.348.2 validation . . . 3766
1.348.3 validation(Validation) . . . 3771
1.349 verdi_neruda . . . 3781
1.349.1 a_star_interpreter(W) . . . 3781
1.349.2 benchmark_generators . . . 3782
1.349.3 best_first . . . 3783
1.349.4 bfs_interpreter . . . 3785
1.349.5 bup_interpreter . . . 3786
1.349.6 counter . . . 3787
1.349.7 databasep . . . 3790
1.349.8 debug_expansion(Mode) . . . 3792
1.349.9 demodb . . . 3793

1.349.10 dfs_interpreter . . . 3794
1.349.11 flatting . . . 3795
1.349.12 heuristic_expansion(Mode) . . . 3796
1.349.13 iddfs_interpreter(Increment) . . . 3797
1.349.14 interpreterp . . . 3798
1.349.15 magic . . . 3800
1.349.16 magic_expansion(Mode) . . . 3801
1.349.17 rule_expansion(Mode) . . . 3802
1.349.18 shell . . . 3803
1.349.19 shell(Interpreters) . . . 3805
1.349.20 shell_expansion(Mode) . . . 3806
1.350 wkt_wkb . . . 3807
1.350.1 wkt_wkb . . . 3807
1.350.2 wkt_wkb_protocol . . . 3808
1.351 wrapper . . . 3812
1.351.1 wrapper . . . 3812
1.352 xml_parser . . . 3823
1.352.1 xml . . . 3823
1.353 yaml . . . 3829
1.353.1 yaml . . . 3829
1.353.2 yaml_protocol . . . 3830
1.354 z_score_anomaly_detector . . . 3833
1.354.1 z_score_anomaly_detector . . . 3834
1.355 zippers . . . 3835
1.355.1 zipperp . . . 3835
1.355.2 zlist . . . 3844

2 Directories **3847**
2.1 contributions/flags/ . . . 3847
2.2 contributions/iso8601/ . . . 3847
2.3 contributions/pddl_parser/ . . . 3847
2.4 contributions/verdi_neruda/ . . . 3847
2.5 contributions/xml_parser/ . . . 3848
2.6 core/ . . . 3848
2.7 library/ . . . 3848
2.8 library/adaptive_boosting_classifier/ . . . 3848
2.9 library/agglomerative_clusterer/ . . . 3848
2.10 library/amqp/ . . . 3848
2.11 library/anomaly_detection_protocols/ . . . 3848
2.12 library/ant_colony/ . . . 3848
2.13 library/application/ . . . 3849
2.14 library/apriori_pattern_miner/ . . . 3849
2.15 library/arbitrary/ . . . 3849
2.16 library/arrangements/ . . . 3849
2.17 library/asp/ . . . 3849
2.18 library/assignvars/ . . . 3849
2.19 library/association_rule_miner/ . . . 3849
2.20 library/atms/ . . . 3850
2.21 library/avro/ . . . 3850
2.22 library/base32/ . . . 3850
2.23 library/base58/ . . . 3850
2.24 library/base64/ . . . 3850
2.25 library/base85/ . . . 3850
2.26 library/baseline_forecasting/ . . . 3850
2.27 library/bayesian_ridge_regression/ . . . 3850
2.28 library/block_ciphers/ . . . 3851
2.29 library/bm25_recommender/ . . . 3851

2.30 library/borda_ranker/ . . . 3851
2.31 library/bradley_terry_ranker/ . . . 3851
2.32 library/bson/ . . . 3851
2.33 library/byte_order/ . . . 3851
2.34 library/c45_classifier/ . . . 3851
2.35 library/cartesian_products/ . . . 3851
2.36 library/cbor/ . . . 3852
2.37 library/ccsds_frames/ . . . 3852
2.38 library/ccsds_link_profiles/ . . . 3852
2.39 library/ccsds_packet_services/ . . . 3852
2.40 library/ccsds_packetization/ . . . 3852
2.41 library/ccsds_packets/ . . . 3852
2.42 library/ccsds_tc_services/ . . . 3852
2.43 library/ccsds_time_codes/ . . . 3852
2.44 library/ccsds_time_fields/ . . . 3853
2.45 library/character_sets/ . . . 3853
2.46 library/chi_square_feature_selector/ . . . 3853
2.47 library/classification_protocols/ . . . 3853
2.48 library/clo_span_pattern_miner/ . . . 3854
2.49 library/clustering_protocols/ . . . 3854
2.50 library/cmac/ . . . 3854
2.51 library/colley_ranker/ . . . 3854
2.52 library/combinations/ . . . 3854
2.53 library/command_line_options/ . . . 3854
2.54 library/constrained_optimization/ . . . 3854
2.55 library/copeland_ranker/ . . . 3855
2.56 library/coroutining/ . . . 3855
2.57 library/crontab/ . . . 3855
2.58 library/crs_projections/ . . . 3855
2.59 library/crypto/ . . . 3855
2.60 library/csv/ . . . 3855
2.61 library/cuid2/ . . . 3855
2.62 library/cusum_anomaly_detector/ . . . 3856
2.63 library/datalog/ . . . 3856
2.64 library/dates/ . . . 3856
2.65 library/dates_tz/ . . . 3856
2.66 library/dbscan_clusterer/ . . . 3856
2.67 library/dependents/ . . . 3856
2.68 library/deques/ . . . 3856
2.69 library/derangements/ . . . 3856
2.70 library/dictionaries/ . . . 3857
2.71 library/dif/ . . . 3857
2.72 library/differential_evolution/ . . . 3857
2.73 library/dimension_reduction_protocols/ . . . 3857
2.74 library/eclat_pattern_miner/ . . . 3857
2.75 library/edcg/ . . . 3857
2.76 library/elastic_net_regression/ . . . 3857
2.77 library/elo_ranker/ . . . 3858
2.78 library/emeans_clusterer/ . . . 3858
2.79 library/events/ . . . 3858
2.80 library/ewma_anomaly_detector/ . . . 3858
2.81 library/expand_library_alias_paths/ . . . 3858
2.82 library/expecteds/ . . . 3858
2.83 library/exponential_smoothing/ . . . 3858
2.84 library/feature_selection_protocols/ . . . 3858
2.85 library/fisher_score_feature_selector/ . . . 3859

2.86 library/flow_networks/ . . . 3859
2.87 library/format/ . . . 3860
2.88 library/fp_growth_pattern_miner/ . . . 3860
2.89 library/frequent_pattern_mining_protocols/ . . . 3860
2.90 library/gaussian_mixture_clusterer/ . . . 3860
2.91 library/gaussian_process_regression/ . . . 3860
2.92 library/genetic_algorithm/ . . . 3860
2.93 library/genint/ . . . 3860
2.94 library/gensym/ . . . 3860
2.95 library/geohash/ . . . 3860
2.96 library/geojson/ . . . 3861
2.97 library/geospatial/ . . . 3861
2.98 library/git/ . . . 3861
2.99 library/glicko2_periodic_ranker/ . . . 3861
2.100 library/glicko2_ranker/ . . . 3861
2.101 library/gpx/ . . . 3861
2.102 library/gradient_boosting_classifier/ . . . 3861
2.103 library/gradient_boosting_regression/ . . . 3861
2.104 library/grammars/ . . . 3861
2.105 library/graphs/ . . . 3862
2.106 library/gravatar/ . . . 3862
2.107 library/gsp_pattern_miner/ . . . 3862
2.108 library/hashes/ . . . 3862
2.109 library/hdbscan_clusterer/ . . . 3863
2.110 library/heaps/ . . . 3863
2.111 library/hierarchical_clustering/ . . . 3864
2.112 library/hierarchies/ . . . 3864
2.113 library/hmac/ . . . 3864
2.114 library/hodge_rank/ . . . 3864
2.115 library/hook_flows/ . . . 3864
2.116 library/hook_objects/ . . . 3864
2.117 library/html/ . . . 3865
2.118 library/http_authenticate/ . . . 3865
2.119 library/http_client/ . . . 3865
2.120 library/http_client_core/ . . . 3865
2.121 library/http_cookies/ . . . 3865
2.122 library/http_core/ . . . 3865
2.123 library/http_cors/ . . . 3866
2.124 library/http_digest/ . . . 3866
2.125 library/http_directory_listing/ . . . 3866
2.126 library/http_htmx/ . . . 3866
2.127 library/http_multipart/ . . . 3866
2.128 library/http_oauth/ . . . 3866
2.129 library/http_parameters/ . . . 3867
2.130 library/http_process_transport/ . . . 3867
2.131 library/http_router/ . . . 3867
2.132 library/http_server/ . . . 3867
2.133 library/http_server_core/ . . . 3867
2.134 library/http_session/ . . . 3867
2.135 library/http_socket_transport/ . . . 3867
2.136 library/http_sse/ . . . 3867
2.137 library/http_static_files/ . . . 3868
2.138 library/http_websocket/ . . . 3868
2.139 library/http_websocket_frames/ . . . 3868
2.140 library/http_websocket_handshake/ . . . 3868
2.141 library/http_websocket_messages/ . . . 3868

2.142 library/http_websocket_service/ . . . 3868
2.143 library/http_websocket_session/ . . . 3868
2.144 library/ica_projection/ . . . 3869
2.145 library/ids/ . . . 3869
2.146 library/ieee_754/ . . . 3869
2.147 library/integer_partitions/ . . . 3869
2.148 library/intermittent_demand_forecasting/ . . . 3869
2.149 library/intervals/ . . . 3869
2.150 library/iqr_anomaly_detector/ . . . 3869
2.151 library/iso_13616/ . . . 3870
2.152 library/iso_3166/ . . . 3870
2.153 library/iso_4217/ . . . 3870
2.154 library/iso_639/ . . . 3870
2.155 library/iso_9362/ . . . 3870
2.156 library/isolation_forest_anomaly_detector/ . . . 3870
2.157 library/jaccard_recommender/ . . . 3870
2.158 library/java/ . . . 3871
2.159 library/json/ . . . 3871
2.160 library/json_graph/ . . . 3871
2.161 library/json_ld/ . . . 3871
2.162 library/json_lines/ . . . 3871
2.163 library/json_patch/ . . . 3871
2.164 library/json_path/ . . . 3872
2.165 library/json_pointer/ . . . 3872
2.166 library/json_rpc/ . . . 3872
2.167 library/json_schema/ . . . 3872
2.168 library/jwt/ . . . 3872
2.169 library/kcenters_clusterer/ . . . 3872
2.170 library/kemeny_young_ranker/ . . . 3873
2.171 library/kernel_pca_projection/ . . . 3873
2.172 library/kernel_svm_classifier/ . . . 3873
2.173 library/kmeans_clusterer/ . . . 3873
2.174 library/kmedians_clusterer/ . . . 3873
2.175 library/kmedoids_clusterer/ . . . 3873
2.176 library/kmodes_clusterer/ . . . 3873
2.177 library/knn_classifier/ . . . 3873
2.178 library/knn_distance_anomaly_detector/ . . . 3873
2.179 library/knn_forecasting/ . . . 3873
2.180 library/knn_item_recommender/ . . . 3874
2.181 library/knn_regression/ . . . 3874
2.182 library/knn_user_recommender/ . . . 3874
2.183 library/kprototypes_clusterer/ . . . 3874
2.184 library/ksuid/ . . . 3874
2.185 library/language_detection/ . . . 3874
2.186 library/language_detection/profiles/ . . . 3874
2.187 library/language_detection/strategies/ . . . 3876
2.188 library/lasso_feature_selector/ . . . 3876
2.189 library/lasso_regression/ . . . 3876
2.190 library/lda_classifier/ . . . 3876
2.191 library/lda_projection/ . . . 3876
2.192 library/lemmatization/ . . . 3876
2.193 library/lemmatization/languages/ . . . 3876
2.194 library/linda/ . . . 3876
2.195 library/linear_algebra/ . . . 3877
2.196 library/linear_programming/ . . . 3877
2.197 library/linear_regression/ . . . 3877

2.198 library/linear_svm_classifier/ . . . 3877
2.199 library/listing/ . . . 3877
2.200 library/local_optimization/ . . . 3877
2.201 library/lof_anomaly_detector/ . . . 3877
2.202 library/logging/ . . . 3878
2.203 library/logistic_regression_classifier/ . . . 3878
2.204 library/loops/ . . . 3878
2.205 library/lru_caches/ . . . 3878
2.206 library/massey_ranker/ . . . 3878
2.207 library/mcp_server/ . . . 3878
2.208 library/memcached/ . . . 3879
2.209 library/message_pack/ . . . 3879
2.210 library/meta/ . . . 3879
2.211 library/meta_compiler/ . . . 3879
2.212 library/mime_types/ . . . 3879
2.213 library/minizinc/ . . . 3879
2.214 library/mlp_classifier/ . . . 3879
2.215 library/modified_z_score_anomaly_detector/ . . . 3879
2.216 library/mqtt/ . . . 3880
2.217 library/mrmr_feature_selector/ . . . 3880
2.218 library/multisets/ . . . 3880
2.219 library/multivariate_distributions/ . . . 3880
2.220 library/mutations/ . . . 3880
2.221 library/mutual_information_feature_selector/ . . . 3880
2.222 library/n_grams/ . . . 3880
2.223 library/naive_bayes_classifier/ . . . 3880
2.224 library/nanoid/ . . . 3881
2.225 library/nearest_centroid_classifier/ . . . 3881
2.226 library/nested_dictionaries/ . . . 3881
2.227 library/nmea/ . . . 3881
2.228 library/nmf_projection/ . . . 3881
2.229 library/numerical_analysis/ . . . 3881
2.230 library/omt/ . . . 3882
2.231 library/one_class_svm_anomaly_detector/ . . . 3882
2.232 library/open_ai/ . . . 3882
2.233 library/open_api/ . . . 3882
2.234 library/open_id/ . . . 3882
2.235 library/optics_clusterer/ . . . 3883
2.236 library/optionals/ . . . 3883
2.237 library/options/ . . . 3883
2.238 library/os/ . . . 3883
2.239 library/otp/ . . . 3883
2.240 library/particle_swarm_optimization/ . . . 3883
2.241 library/partitions/ . . . 3883
2.242 library/paseto/ . . . 3883
2.243 library/pattern_mining_protocols/ . . . 3884
2.244 library/pca_projection/ . . . 3884
2.245 library/permutations/ . . . 3884
2.246 library/plackett_luce_last_ranker/ . . . 3884
2.247 library/plackett_luce_ranker/ . . . 3884
2.248 library/pls_projection/ . . . 3884
2.249 library/prefix_span_pattern_miner/ . . . 3884
2.250 library/probabilistic_pca_projection/ . . . 3884
2.251 library/problog/ . . . 3884
2.252 library/process/ . . . 3885
2.253 library/protobuf/ . . . 3885

2.254 library/qda_classifier/ . . . 3885
2.255 library/queues/ . . . 3885
2.256 library/random/ . . . 3885
2.257 library/random_forest_classifier/ . . . 3885
2.258 library/random_forest_regression/ . . . 3885
2.259 library/random_projection/ . . . 3885
2.260 library/rank_centrality/ . . . 3886
2.261 library/ranked_pairs/ . . . 3886
2.262 library/ranking_protocols/ . . . 3886
2.263 library/reader/ . . . 3886
2.264 library/recommender_protocols/ . . . 3886
2.265 library/recorded_database/ . . . 3887
2.266 library/redis/ . . . 3887
2.267 library/regression_protocols/ . . . 3887
2.268 library/regression_tree/ . . . 3887
2.269 library/regularized_bradley_terry_ranker/ . . . 3887
2.270 library/relief_feature_selector/ . . . 3887
2.271 library/relieff_feature_selector/ . . . 3887
2.272 library/rest/ . . . 3887
2.273 library/ridge_regression/ . . . 3887
2.274 library/ripple_down_rules/ . . . 3888
2.275 library/rrelieff_feature_selector/ . . . 3888
2.276 library/s3/ . . . 3888
2.277 library/sat/ . . . 3888
2.278 library/schulze_ranker/ . . . 3888
2.279 library/sequential_pattern_mining_protocols/ . . . 3888
2.280 library/sets/ . . . 3889
2.281 library/sgd_classifier/ . . . 3889
2.282 library/simple_temporal_networks/ . . . 3889
2.283 library/simulated_annealing/ . . . 3889
2.284 library/slope_one_recommender/ . . . 3889
2.285 library/smt/ . . . 3889
2.286 library/smtp/ . . . 3889
2.287 library/snowflakeid/ . . . 3890
2.288 library/sockets/ . . . 3890
2.289 library/souffle/ . . . 3890
2.290 library/spade_pattern_miner/ . . . 3890
2.291 library/spectral_clusterer/ . . . 3890
2.292 library/sqids/ . . . 3890
2.293 library/statistics/ . . . 3890
2.294 library/stemming/ . . . 3891
2.295 library/stomp/ . . . 3891
2.296 library/stop_words/ . . . 3891
2.297 library/stop_words/languages/ . . . 3891
2.298 library/string_distance/ . . . 3892
2.299 library/strings/ . . . 3892
2.300 library/subsequences/ . . . 3892
2.301 library/svr_regression/ . . . 3893
2.302 library/tabu_search/ . . . 3893
2.303 library/term_io/ . . . 3893
2.304 library/text_normalization/ . . . 3893
2.305 library/text_normalization/data/ . . . 3893
2.306 library/text_normalization/profiles/ . . . 3893
2.307 library/text_vectorization/ . . . 3893
2.308 library/tfidf_recommender/ . . . 3894
2.309 library/theta_forecasting/ . . . 3894

2.310 library/thurstone_mosteller_ranker/ . . . 3894
2.311 library/time_scales/ . . . 3894
2.312 library/time_series_protocols/ . . . 3894
2.313 library/time_series_regression/ . . . 3894
2.314 library/timeout/ . . . 3894
2.315 library/tle_orbits/ . . . 3894
2.316 library/tokenization/ . . . 3894
2.317 library/tokenization/languages/ . . . 3895
2.318 library/toml/ . . . 3895
2.319 library/toon/ . . . 3895
2.320 library/tries/ . . . 3895
2.321 library/trueskill_ranker/ . . . 3895
2.322 library/truncated_svd_projection/ . . . 3895
2.323 library/tsne_projection/ . . . 3895
2.324 library/tsv/ . . . 3895
2.325 library/typeid/ . . . 3896
2.326 library/types/ . . . 3896
2.327 library/tzif/ . . . 3896
2.328 library/ulid/ . . . 3896
2.329 library/umap_projection/ . . . 3897
2.330 library/union_find/ . . . 3897
2.331 library/univariate_distributions/ . . . 3897
2.332 library/uri_template/ . . . 3897
2.333 library/url/ . . . 3897
2.334 library/uuid/ . . . 3897
2.335 library/validations/ . . . 3897
2.336 library/wkt_wkb/ . . . 3897
2.337 library/yaml/ . . . 3898
2.338 library/z_score_anomaly_detector/ . . . 3898
2.339 library/zippers/ . . . 3898
2.340 ports/fcube/ . . . 3898
2.341 ports/metagol/ . . . 3898
2.342 ports/toychr/ . . . 3898
2.343 tools/assertions/ . . . 3898
2.344 tools/code_metrics/ . . . 3898
2.345 tools/dead_code_scanner/ . . . 3899
2.346 tools/debug_messages/ . . . 3899
2.347 tools/debugger/ . . . 3899
2.348 tools/diagrams/ . . . 3899
2.349 tools/doclet/ . . . 3900
2.350 tools/help/ . . . 3900
2.351 tools/issue_creator/ . . . 3900
2.352 tools/lgtdoc/ . . . 3900
2.353 tools/lgtunit/ . . . 3900
2.354 tools/linter_reporter/ . . . 3901
2.355 tools/mutation_testing/ . . . 3901
2.356 tools/mutation_testing/mutators/ . . . 3901
2.357 tools/packs/ . . . 3902
2.358 tools/ports_profiler/ . . . 3902
2.359 tools/sarif/ . . . 3902
2.360 tools/sbom/ . . . 3902
2.361 tools/tool_diagnostics/ . . . 3902
2.362 tools/tutor/ . . . 3902
2.363 tools/wrapper/ . . . 3902

3 Entities **3903**
3.1 Categories . . . 3903

3.2 Objects 3906
3.3 Protocols 3925

4 Predicates **3931**

Index **4001**

CHAPTER
ONE

LIBRARIES

To load any library (including developer tools, ports, and contributions), use the goal `logtalk_load(library_name(loader))`. To run the library tests, use the goal `logtalk_load(library_name(tester))`. To load an entity, always load the loader file of the library that includes it to ensure that all required dependencies are also loaded and that any required flags are used. The loading goal can be found in the entity documentation.

1.1 adaptive_boosting_classifier

object

1.1.1 `adaptive_boosting_classifier`

Adaptive Boosting classifier using C4.5 decision trees as base learners. Implements the SAMME (Stagewise Additive Modeling using a Multi-class Exponential loss function) variant, which supports multi-class classification. Builds an ensemble of weighted decision trees where each subsequent tree focuses on the examples misclassified by previous trees.

Availability:
`logtalk_load(adaptive_boosting_classifier(loader))`

Author: Paulo Moura
Version: 3:1:0
Date: 2026-10-29

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *probabilistic_classifier_common*
Uses:
avltree
c45_classifier
fast_random(Algorithm)
format
list
pairs
type

Remarks:
(none)

Inherited public predicates:

check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *check_ensemble/2*
  - *train_with_cleanup/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

check_ensemble/2

Checks that learning accepted at least one estimator.

Compilation flags:

```
static
```

Template:

```
check_ensemble(Dataset,WeightedTrees)
```

Mode and number of proofs:

```
check_ensemble(+object_identifier,+list) - one_or_error
```

Exceptions:

No estimator was accepted:

```
domain_error(non_empty_ensemble,Dataset)
```

---

train_with_cleanup/1

Learns a C4.5 tree using a dataset-tree pair and abolishes the temporary dataset on success, failure, or exception.

Compilation flags:

```
static
```

Template:

```
train_with_cleanup(DatasetTree)
```

Mode and number of proofs:

`train_with_cleanup(+pair(object_identifier,term))` - `zero_or_one`

Exceptions:

C4.5 learning throws an exception:

`Exception`

Operators

(none)

See also

dataset_protocol, *c45_classifier*, *isolation_forest_anomaly_detector*, *knn_classifier*, *naive_bayes_classifier*, *nearest_centroid_classifier*, *random_forest_classifier*

1.2 agglomerative_clusterer

object

1.2.1 `agglomerative_clusterer`

Agglomerative clusterer for continuous datasets. Learns from a dataset object implementing the `clustering_dataset_protocol` protocol and returns a clusterer term that can be used for assigning new instances to clusters and exported as predicate clauses.

Availability:

`logtalk_load(agglomerative_clusterer(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *clusterer_common*

Uses:

avltree
binary_heap_min
format
list
numberlist
type

Remarks:

(none)

Inherited public predicates:

check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *kcenters_clusterer*, *kmeans_clusterer*, *kmedoids_clusterer*

1.3 amqp

object

1.3.1 `amqp`

Portable AMQP 0-9-1 (Advanced Message Queuing Protocol) client. Uses the sockets library for TCP communication.

Availability:

```
logtalk_load(amqp(loader))
```

Author: Paulo Moura
Version: 1:4:0
Date: 2026-10-23

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public options
```

Uses:

```
byte_order
list
os
socket
term_io
```

Remarks:

- Supported backends: ECLiPSe, GNU Prolog, SICStus Prolog, SWI-Prolog, Trealla Prolog, and XVM (same as the sockets library).
- Protocol version: Implements AMQP 0-9-1 specification.
- Binary protocol: AMQP is a binary protocol with typed frame encoding.
- Channels: Supports multiple concurrent channels over a single connection.
- Exchanges and queues: Full support for declaring exchanges, queues, and bindings.
- Content: Supports message publishing and consuming with content headers.
- Transactions: Supports AMQP transactions with tx.select, tx.commit, and tx.rollback.
- Publisher confirms: Support for publisher confirms can be added.
- Heartbeat: Supports heartbeat negotiation to keep connections alive.
- Reconnection: Automatic reconnection with configurable retry attempts and delays.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *connect/4*
  - *close/1*
  - *close/3*
  - *connection_alive/1*
  - *channel_open/3*
  - *channel_close/1*
  - *channel_close/3*
  - *exchange_declare/3*
  - *exchange_delete/3*
  - *exchange_bind/4*
  - *exchange_unbind/4*
  - *queue_declare/3*
  - *queue_delete/3*
  - *queue_bind/4*
  - *queue_unbind/4*
  - *queue_purge/2*
  - *basic_publish/4*
  - *basic_consume/3*

- basic_cancel/3
- basic_get/3
- basic_ack/3
- basic_nack/3
- basic_reject/3
- basic_qos/2
- basic_recover/2
- receive/3
- tx_select/1
- tx_commit/1
- tx_rollback/1
- confirm_select/1
- send_heartbeat/1
- message_body/2
- message_properties/2
- message_property/3
- message_delivery_tag/2
- message_exchange/2
- message_routing_key/2
- encode_frame/2
- decode_frame/2

- *Protected predicates*
- *Private predicates*
  - decode_field_value/3
- *Operators*

Public predicates

connect/4

Connects to an AMQP 0-9-1 server and performs the protocol handshake. Returns a connection handle for subsequent operations. Supports automatic reconnection on connection failures.

Compilation flags:

```
static
```

Template:

```
connect(Host,Port,Connection,Options)
```

Mode and number of proofs:

```
connect(+atom,+integer,--compound,+list) - one_or_error
```

Exceptions:

Connection refused or network error:

```
amqp_error(connection_failed)
```

Server rejected connection:

```
amqp_error(protocol_error(Message))
```

Authentication failed:

```
amqp_error(auth_failed)
```

All reconnection attempts failed:
`amqp_error(reconnect_failed)`

Remarks:

- Option username(Username): Username for authentication. Default is guest.
- Option password(Password): Password for authentication. Default is guest.
- Option virtual_host(VHost): Virtual host name. Default is /.
- Option heartbeat(Seconds): Heartbeat interval in seconds. Default is 60.
- Option channel_max(Max): Maximum number of channels. Default is 0 (no limit).
- Option frame_max(Max): Maximum frame size. Default is 131072.
- Option reconnect(Boolean): Enable automatic reconnection on connection failure. Default is false.
- Option reconnect_attempts(N): Maximum number of reconnection attempts. Default is 3. Only used when reconnect(true).
- Option reconnect_delay(Seconds): Delay between reconnection attempts in seconds. Default is 1. Only used when reconnect(true).

`close/1`

Gracefully closes the AMQP connection. Closes all channels and the connection itself.

Compilation flags:
`static`

Template:
`close(Connection)`
Mode and number of proofs:
`close(+compound)` - `one_or_error`

`close/3`

Closes the AMQP connection with a specific reply code and reason.

Compilation flags:
`static`

Template:
`close(Connection,ReplyCode,ReplyText)`
Mode and number of proofs:
`close(+compound,+integer,+atom)` - `one_or_error`

Exceptions:
`Connection` is not a valid AMQP connection handle:
`error`

---

connection_alive/1

Checks if the connection is still open and valid.

Compilation flags:
`static`

Template:
`connection_alive(Connection)`
Mode and number of proofs:
`connection_alive(+compound)` - `zero_or_one`

---

channel_open/3

Opens a new channel on the connection. Returns a channel handle.

Compilation flags:
`static`

Template:
`channel_open(Connection,ChannelNumber,Channel)`
Mode and number of proofs:
`channel_open(+compound,+integer,--compound)` - `one_or_error`

Exceptions:
Channel already open:
`amqp_error(channel_error(Message))`

---

channel_close/1

Closes a channel.

Compilation flags:
`static`

Template:
`channel_close(Channel)`
Mode and number of proofs:
`channel_close(+compound)` - `one_or_error`

Exceptions:
`Channel` is not a valid AMQP channel handle:
`error`

---

channel_close/3

Closes a channel with a specific reply code and reason.

Compilation flags:
`static`

Template:
`channel_close(Channel,ReplyCode,ReplyText)`
Mode and number of proofs:
`channel_close(+compound,+integer,+atom)` - `one_or_error`

Exceptions:
`Channel` is not a valid AMQP channel handle:
`error`

---

exchange_declare/3

Declares an exchange on the server.

Compilation flags:
`static`

Template:
`exchange_declare(Channel,Exchange,Options)`
Mode and number of proofs:
`exchange_declare(+compound,+atom,+list)` - `one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the exchange declaration:
`amqp_error(exchange_error(Message))`

Remarks:

- Option type(Type): Exchange type: direct, fanout, topic, headers. Default is direct.
- Option durable(Boolean): Survive server restart. Default is false.
- Option auto_delete(Boolean): Delete when unused. Default is false.
- Option internal(Boolean): Internal exchange. Default is false.
- Option arguments(Arguments): Additional arguments as key-value pairs.

---

exchange_delete/3

Deletes an exchange.

Compilation flags:
`static`

Template:
`exchange_delete(Channel,Exchange,Options)`
Mode and number of proofs:
`exchange_delete(+compound,+atom,+list)` - `one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the exchange deletion:
`amqp_error(exchange_error(Message))`

Remarks:

- Option if_unused(Boolean): Only delete if unused. Default is false.

---

exchange_bind/4

Binds an exchange to another exchange.

Compilation flags:
`static`

Template:
`exchange_bind(Channel,Destination,Source,Options)`
Mode and number of proofs:
`exchange_bind(+compound,+atom,+atom,+list)` - `one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the exchange binding:
`amqp_error(exchange_error(Message))`

Remarks:

- Option routing_key(Key): Routing key for binding. Default is empty.
- Option arguments(Arguments): Additional arguments.

---

exchange_unbind/4

Unbinds an exchange from another exchange.

Compilation flags:
`static`

Template:
`exchange_unbind(Channel,Destination,Source,Options)`
Mode and number of proofs:
`exchange_unbind(+compound,+atom,+atom,+list)` - `one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the exchange unbinding:
`amqp_error(exchange_error(Message))`

---

queue_declare/3

Declares a queue on the server. If Queue is a variable, the server generates a unique name.

Compilation flags:
`static`

Template:
`queue_declare(Channel,Queue,Options)`
Mode and number of proofs:
`queue_declare(+compound,?atom,+list)` - `one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the queue declaration:
`amqp_error(queue_error(Message))`

Remarks:

- Option durable(Boolean): Survive server restart. Default is false.
- Option exclusive(Boolean): Exclusive to this connection. Default is false.
- Option auto_delete(Boolean): Delete when unused. Default is false.
- Option arguments(Arguments): Additional arguments (e.g., message TTL, dead letter exchange).

---

queue_delete/3

Deletes a queue.

Compilation flags:
`static`

Template:
`queue_delete(Channel,Queue,Options)`
Mode and number of proofs:
`queue_delete(+compound,+atom,+list) - one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the queue deletion:
`amqp_error(queue_error(Message))`

Remarks:

- Option if_unused(Boolean): Only delete if unused. Default is false.
- Option if_empty(Boolean): Only delete if empty. Default is false.

---

queue_bind/4

Binds a queue to an exchange.

Compilation flags:
`static`

Template:
`queue_bind(Channel,Queue,Exchange,Options)`
Mode and number of proofs:
`queue_bind(+compound,+atom,+atom,+list) - one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the queue binding:
`amqp_error(queue_error(Message))`

Remarks:

- Option routing_key(Key): Routing key for binding. Default is empty.
- Option arguments(Arguments): Additional arguments.

---

queue_unbind/4

Unbinds a queue from an exchange.

Compilation flags:
`static`

Template:
`queue_unbind(Channel,Queue,Exchange,Options)`
Mode and number of proofs:
`queue_unbind(+compound,+atom,+atom,+list) - one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the queue unbinding:
`amqp_error(queue_error(Message))`

---

queue_purge/2

Purges all messages from a queue.

Compilation flags:
`static`

Template:
`queue_purge(Channel,Queue)`
Mode and number of proofs:
`queue_purge(+compound,+atom) - one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the queue purge request:
`amqp_error(queue_error(Message))`

---

basic_publish/4

Publishes a message to an exchange.

Compilation flags:
`static`

Template:
`basic_publish(Channel,Exchange,Body,Options)`
Mode and number of proofs:
`basic_publish(+compound,+atom,+term,+list) - one_or_error`

Exceptions:

`Channel` is invalid, `Body` cannot be encoded, or the frame/content transmission fails:
`error`

Remarks:

- Option routing_key(Key): Routing key for message. Default is empty.
- Option mandatory(Boolean): Return if not routable. Default is false.
- Option immediate(Boolean): Return if not deliverable. Default is false (deprecated in RabbitMQ).
- Option content_type(Type): MIME content type.
- Option content_encoding(Enc): Content encoding.
- Option correlation_id(Id): Correlation identifier.
- Option reply_to(Queue): Reply queue name.
- Option expiration(Ms): Message TTL in milliseconds.
- Option message_id(Id): Application message identifier.
- Option timestamp(Ts): Message timestamp.
- Option type(Type): Message type name.
- Option user_id(Id): Creating user ID.
- Option app_id(Id): Creating application ID.
- Option delivery_mode(Mode): 1 for non-persistent, 2 for persistent.
- Option priority(P): Message priority (0-9).
- Option headers(H): Application headers as key-value pairs.

`basic_consume/3`

Starts consuming messages from a queue.

Compilation flags:
`static`

Template:
`basic_consume(Channel,Queue,Options)`
Mode and number of proofs:
`basic_consume(+compound,+atom,+list) - one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the consume request:
`amqp_error(basic_error(Message))`

Remarks:

- Option consumer_tag(Tag): Consumer identifier. Server generates if not provided.
- Option no_local(Boolean): Do not receive own messages. Default is false.
- Option no_ack(Boolean): No acknowledgment required. Default is false.
- Option exclusive(Boolean): Exclusive consumer. Default is false.
- Option arguments(Arguments): Additional arguments.

---

basic_cancel/3

Cancels a consumer.

Compilation flags:
`static`

Template:
`basic_cancel(Channel,ConsumerTag,Options)`
Mode and number of proofs:
`basic_cancel(+compound,+atom,+list)` - `one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the cancel request:
`amqp_error(basic_error(Message))`

---

basic_get/3

Synchronously gets a message from a queue.

Compilation flags:
`static`

Template:
`basic_get(Channel,Queue,Options)`
Mode and number of proofs:
`basic_get(+compound,+atom,+list)` - `one_or_error`

Exceptions:
`Channel` is invalid or the server replies with an unexpected `basic.get` result:
`amqp_error(basic_error(Message))`

Remarks:

- Option no_ack(Boolean): No acknowledgment required. Default is false.

---

basic_ack/3

Acknowledges a message.

Compilation flags:
`static`

Template:
`basic_ack(Channel,DeliveryTag,Options)`
Mode and number of proofs:

`basic_ack(+compound,+integer,+list)` - `one_or_error`

Exceptions:

`Channel` is invalid or the acknowledgment frame cannot be sent:

`error`

Remarks:

- Option multiple(Boolean): Acknowledge all up to this tag. Default is false.

---

basic_nack/3

Negatively acknowledges a message (RabbitMQ extension).

Compilation flags:

`static`

Template:

`basic_nack(Channel,DeliveryTag,Options)`

Mode and number of proofs:

`basic_nack(+compound,+integer,+list)` - `one_or_error`

Exceptions:

`Channel` is invalid or the negative-acknowledgment frame cannot be sent:

`error`

Remarks:

- Option multiple(Boolean): Reject all up to this tag. Default is false.
- Option requeue(Boolean): Requeue the message. Default is true.

---

basic_reject/3

Rejects a message.

Compilation flags:

`static`

Template:

`basic_reject(Channel,DeliveryTag,Options)`

Mode and number of proofs:

`basic_reject(+compound,+integer,+list)` - `one_or_error`

Exceptions:

`Channel` is invalid or the reject frame cannot be sent:

`error`

Remarks:

- Option requeue(Boolean): Requeue the message. Default is true.

---

basic_qos/2

Sets quality of service (prefetch) settings.

Compilation flags:
`static`

Template:
`basic_qos(Channel,Options)`
Mode and number of proofs:
`basic_qos(+compound,+list)` - `one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the QoS request:
`amqp_error(basic_error(Message))`

Remarks:

- Option prefetch_size(Size): Prefetch window size in bytes. Default is 0 (no limit).
- Option prefetch_count(Count): Prefetch window in messages. Default is 0 (no limit).
- Option global(Boolean): Apply to entire connection. Default is false.

---

basic_recover/2

Asks the server to redeliver unacknowledged messages.

Compilation flags:
`static`

Template:
`basic_recover(Channel,Options)`
Mode and number of proofs:
`basic_recover(+compound,+list)` - `one_or_error`

Exceptions:
`Channel` is invalid or the server rejects the recover request:
`amqp_error(basic_error(Message))`

Remarks:

- Option requeue(Boolean): Requeue messages. Default is false.

---

receive/3

Receives a message or method from the server. Blocks until data is available or timeout.

Compilation flags:
`static`

Template:
`receive(Channel,Message,Options)`
Mode and number of proofs:
`receive(+compound,-compound,+list)` - `zero_or_one_or_error`

Exceptions:
`Channel` is invalid or the received frame sequence is not a valid AMQP content delivery:
`error`

Remarks:

- Option timeout(Milliseconds): Timeout in milliseconds. 0 for non-blocking, -1 for infinite. Default is -1.

---

tx_select/1

Enables transaction mode on a channel.

Compilation flags:
`static`

Template:
`tx_select(Channel)`
Mode and number of proofs:
`tx_select(+compound)` - `one_or_error`

Exceptions:
`Channel` is invalid or the server rejects `tx.select`:
`amqp_error(tx_error(Message))`

---

tx_commit/1

Commits the current transaction.

Compilation flags:
`static`

Template:
`tx_commit(Channel)`
Mode and number of proofs:

`tx_commit(+compound)` - `one_or_error`

Exceptions:

`Channel` is invalid or the server rejects `tx.commit`:

`amqp_error(tx_error(Message))`

`tx_rollback/1`

Rolls back the current transaction.

Compilation flags:

`static`

Template:

`tx_rollback(Channel)`

Mode and number of proofs:

`tx_rollback(+compound)` - `one_or_error`

Exceptions:

`Channel` is invalid or the server rejects `tx.rollback`:

`amqp_error(tx_error(Message))`

`confirm_select/1`

Enables publisher confirms on a channel (RabbitMQ extension).

Compilation flags:

`static`

Template:

`confirm_select(Channel)`

Mode and number of proofs:

`confirm_select(+compound)` - `one_or_error`

Exceptions:

`Channel` is invalid or the server rejects `confirm.select`:

`amqp_error(confirm_error(Message))`

---

send_heartbeat/1

Sends a heartbeat frame to the server.

Compilation flags:
`static`

Template:
`send_heartbeat(Connection)`
Mode and number of proofs:
`send_heartbeat(+compound)` - `one_or_error`

Exceptions:
`Connection` is not a valid AMQP connection handle or the heartbeat frame cannot be sent:
`error`

---

message_body/2

Extracts the body from a message.

Compilation flags:
`static`

Template:
`message_body(Message,Body)`
Mode and number of proofs:
`message_body(+compound,-term)` - `one`

---

message_properties/2

Extracts the properties from a message as a list.

Compilation flags:
`static`

Template:
`message_properties(Message,Properties)`
Mode and number of proofs:
`message_properties(+compound,-list)` - `one`

---

message_property/3

Extracts a specific property from a message. Fails if not present.

Compilation flags:
```
static
```

Template:
```
message_property(Message,PropertyName,Value)
```
Mode and number of proofs:
```
message_property(+compound,+atom,-term) - zero_or_one
```

---

message_delivery_tag/2

Extracts the delivery tag from a message.

Compilation flags:
```
static
```

Template:
```
message_delivery_tag(Message,DeliveryTag)
```
Mode and number of proofs:
```
message_delivery_tag(+compound,-integer) - one
```

---

message_exchange/2

Extracts the exchange name from a message.

Compilation flags:
```
static
```

Template:
```
message_exchange(Message,Exchange)
```
Mode and number of proofs:
```
message_exchange(+compound,-atom) - one
```

---

message_routing_key/2

Extracts the routing key from a message.

Compilation flags:
`static`

Template:
`message_routing_key(Message,RoutingKey)`
Mode and number of proofs:
`message_routing_key(+compound,-atom)` - `one`

---

encode_frame/2

Encodes an AMQP frame to a list of bytes.

Compilation flags:
`static`

Template:
`encode_frame(Frame,Bytes)`
Mode and number of proofs:
`encode_frame(+compound,-list)` - `one`

---

decode_frame/2

Decodes a list of bytes to an AMQP frame.

Compilation flags:
`static`

Template:
`decode_frame(Bytes,Frame)`
Mode and number of proofs:
`decode_frame(+list,-compound)` - `one_or_error`

Exceptions:
`Bytes` do not encode a valid complete AMQP frame payload:
`error`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`decode_field_value/3`

Decodes one AMQP field-table value from a byte list, returning the normalized typed value plus the remaining undecoded bytes.

Compilation flags:
`static`

Template:
`decode_field_value(Bytes,Value,Rest)`
Mode and number of proofs:
`decode_field_value(+list(integer),-compound,-list(integer))` - `one_or_error`

Exceptions:
`Bytes` do not start with a supported AMQP field-table value encoding:
`error`

Operators

(none)
category

1.3.2 `amqp_pool`

AMQP connection pool category. Import this category into an object to create a named connection pool with automatic connection management.

Availability:
`logtalk_load(amqp(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-06-16

Compilation flags:
`static`

Extends:
`public` *options*
Uses:
amqp
list
os

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *initialize/1*
  - *destroy/0*
  - *acquire/1*
  - *release/1*
  - *with_connection/1*
  - *stats/1*
  - *resize/2*
- *Protected predicates*
- *Private predicates*
  - *pool_config/5*
  - *available/1*
  - *in_use/2*
- *Operators*

Public predicates

initialize/1

Initializes the connection pool with the given configuration options. Must be called before using other pool predicates.

Compilation flags:
`static`

Template:
`initialize(Options)`
Mode and number of proofs:
`initialize(+list)` - `one_or_error`

Exceptions:
`Options` configure a minimum pool size greater than the maximum pool size:
`pool_error(invalid_config(min_size > max_size))`

Remarks:

- Option host(Host): AMQP server hostname. Default is localhost.
- Option port(Port): AMQP server port. Default is 5672.
- Option min_size(N): Minimum number of connections to maintain. Default is 1.
- Option max_size(N): Maximum number of connections allowed. Default is 10.
- Option connection_options(Options): Options passed to amqp::connect/4. Default is [].

destroy/0

Destroys the pool, closing all connections and clearing state.

Compilation flags:
`static`

Mode and number of proofs:
`destroy` - `one`

---

acquire/1

Acquires a connection from the pool. Returns an available connection or creates a new one if the pool is not at maximum capacity.

Compilation flags:
`static`

Template:
`acquire(Connection)`
Mode and number of proofs:
`acquire(--compound)` - `one_or_error`

Exceptions:
Pool not initialized:
`pool_error(not_initialized)`
Pool exhausted (at max capacity):
`pool_error(exhausted)`

---

release/1

Releases a connection back to the pool, making it available for reuse.

Compilation flags:
`static`

Template:
`release(Connection)`
Mode and number of proofs:
`release(+compound)` - `one`

---

with_connection/1

Acquires a connection, calls Goal with the connection as argument, and releases the connection. The connection is released even if Goal fails or throws an exception.

Compilation flags:
`static`

Template:
`with_connection(Goal)`
Meta-predicate template:
`with_connection(1)`
Mode and number of proofs:
`with_connection(+callable)` - `zero_or_more`

---

stats/1

Returns pool statistics as a compound term stats(Available, InUse, Total, MinSize, MaxSize).

Compilation flags:
`static`

Template:
`stats(Stats)`
Mode and number of proofs:
`stats(-compound)` - `one`

---

resize/2

Resizes the pool by setting new minimum and maximum sizes.

Compilation flags:
`static`

Template:
`resize(MinSize,MaxSize)`
Mode and number of proofs:
`resize(+integer,+integer)` - `one_or_error`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

pool_config/5

Stores the pool configuration parameters.

Compilation flags:
`dynamic`

Template:
`pool_config(Host,Port,MinSize,MaxSize,ConnectionOptions)`
Mode and number of proofs:
`pool_config(?atom,?integer,?integer,?integer,?list)` - `zero_or_one`

---

available/1

Tracks connections that are available for use.

Compilation flags:
`dynamic`

Template:
`available(Connection)`
Mode and number of proofs:
`available(?compound)` - `zero_or_more`

---

in_use/2

Tracks connections currently in use along with their acquisition timestamp.

Compilation flags:
`dynamic`

Template:
`in_use(Connection,AcquireTimestamp)`
Mode and number of proofs:
`in_use(?compound,?compound)` - `zero_or_more`

Operators

(none)

1.4 anomaly_detection_protocols

protocol

1.4.1 `anomaly_dataset_protocol`

Protocol for datasets used with anomaly-detection algorithms.

Availability:
`logtalk_load(anomaly_detection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-19

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *attribute_values/2*
  - *class/1*
  - *class_values/1*
  - *example/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

attribute_values/2

Enumerates by backtracking the attributes and their possible values. For discrete attributes, `Values` is a list of possible values. For continuous attributes, `Values` is the atom `continuous`.

Compilation flags:

```
static
```

Template:

```
attribute_values(Attribute,Values)
```

Mode and number of proofs:

```
attribute_values(?atom,-list(atom)) - zero_or_more
attribute_values(?atom,-atom) - zero_or_more
```

---

class/1

Returns the name of the target label attribute.

Compilation flags:

```
static
```

Template:

```
class(Class)
```

Mode and number of proofs:

```
class(-atom) - one
```

---

class_values/1

Returns the list of possible labels for the target attribute. Current anomaly benchmarks conventionally use `[normal, anomaly]`.

Compilation flags:

```
static
```

Template:

```
class_values(Values)
```

Mode and number of proofs:

```
class_values(-list(atom)) - one
```

---

example/3

Enumerates by backtracking the examples in the dataset. Each example has an `Id`, a `Class` label, and a list of `Attribute-Value` pairs.

Compilation flags:
`static`

Template:
`example(Id,Class,AttributeValues)`
Mode and number of proofs:
`example(-integer,-atom,-list(pair))` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.4.2 anomaly_detector_common

Shared predicates for anomaly detector learning defaults, threshold-based prediction, baseline training-data selection, dataset helpers, and export.

Availability:
`logtalk_load(anomaly_detection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-01

Compilation flags:
`static`

Implements:
`public` *anomaly_detector_protocol*
Extends:
`public` *options*
Uses:
format
list
type

Remarks:

(none)

Inherited public predicates:

anomaly_detector_options/2 *check_anomaly_detector/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict/4* *print_anomaly_detector/1* *score/3* *score_all/3* *valid_anomaly_detector/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *anomaly_detector_diagnostics_data/2*
  - *anomaly_detector_export_template/2*
  - *anomaly_detector_term_template/2*
  - *dataset_attributes/2*
  - *check_examples_non_empty/2*
  - *baseline_training_examples/3*
  - *valid_baseline_class_values/1*
  - *extract_scores/2*
  - *print_anomaly_detector_template/1*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

anomaly_detector_diagnostics_data/2

Hook predicate that importing anomaly detector implementations may override in order to expose diagnostics and metadata for learned detectors.

Compilation flags:

```
static
```

Template:

```
anomaly_detector_diagnostics_data(AnomalyDetector,Diagnostics)
```

Mode and number of proofs:

```
anomaly_detector_diagnostics_data(+compound,-list(compound)) - one
```

anomaly_detector_export_template/2

Hook predicate that importing anomaly detector implementations must define in order to expose the exported detector predicate template for a given functor.

Compilation flags:

```
static
```

Template:

```
anomaly_detector_export_template(Functor,Template)
```

Mode and number of proofs:

```
anomaly_detector_export_template(+atom,-callable) - one
```

---

anomaly_detector_term_template/2

Hook predicate that importing anomaly detector implementations must define in order to expose the learned detector term template used by pretty-printing helpers.

Compilation flags:

```
static
```

Template:

```
anomaly_detector_term_template(AnomalyDetector,Template)
```

Mode and number of proofs:

```
anomaly_detector_term_template(+compound,-compound) - one
```

---

dataset_attributes/2

Collects the dataset attribute declarations as *Attribute-Values* pairs.

Compilation flags:

```
static
```

Template:

```
dataset_attributes(Dataset,Attributes)
```

Mode and number of proofs:

```
dataset_attributes(+object_identifier,-list(pair)) - one
```

---

check_examples_non_empty/2

Checks that a dataset contains at least one example.

Compilation flags:
`static`

Template:
`check_examples_non_empty(Dataset,Examples)`
Mode and number of proofs:
`check_examples_non_empty(+object_identifier,+list)` - `one`

---

baseline_training_examples/3

Collects training examples selected by the `baseline_class_values/1` and `baseline_selection_policy/1` options. The `reject` policy throws an error when non-baseline examples are present. The `filter` policy removes non-baseline examples and throws an error when no baseline examples remain.

Compilation flags:
`static`

Template:
`baseline_training_examples(Dataset,Examples,Options)`
Mode and number of proofs:
`baseline_training_examples(+object_identifier,-list,+list(compound))` - `one`

---

valid_baseline_class_values/1

Checks that a `baseline_class_values/1` option value is a non-empty list of distinct atoms.

Compilation flags:
`static`

Template:
`valid_baseline_class_values(BaselineClassValues)`
Mode and number of proofs:
`valid_baseline_class_values(+list(atom))` - `one`

---

extract_scores/2

Transforms *Score-Id-Class* tuples into the public *Id-Class-Score* representation.

Compilation flags:
```
static
```

Template:
```
extract_scores(Pairs,Scores)
```
Mode and number of proofs:
```
extract_scores(+list,-list) - one
```

---

print_anomaly_detector_template/1

Prints the learned anomaly detector term template in a human-readable form.

Compilation flags:
```
static
```

Template:
```
print_anomaly_detector_template(AnomalyDetector)
```
Mode and number of proofs:
```
print_anomaly_detector_template(+compound) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.4.3 anomaly_detector_protocol

Protocol for machine learning anomaly detectors.

Availability:
```
logtalk_load(anomaly_detection_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
```
static
```

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/2*
  - *learn/3*
  - *check_anomaly_detector/1*
  - *valid_anomaly_detector/1*
  - *diagnostics/2*
  - *diagnostic/2*
  - *anomaly_detector_options/2*
  - *predict/3*
  - *predict/4*
  - *score/3*
  - *score_all/3*
  - *export_to_clauses/4*
  - *export_to_file/4*
  - *print_anomaly_detector/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/2

Learns an anomaly detector from the given dataset object.

Compilation flags:
`static`

Template:
`learn(Dataset,AnomalyDetector)`

Mode and number of proofs:
`learn(+object_identifier,-compound)` - `one`

learn/3

Learns an anomaly detector from the given dataset object using the specified options.

Compilation flags:
`static`

Template:
`learn(Dataset,AnomalyDetector,Options)`
Mode and number of proofs:
`learn(+object_identifier,-compound,+list(compound))` - `one`

---

check_anomaly_detector/1

Checks that a learned anomaly detector term is structurally valid for the receiving implementation. Throws an exception when the term is not a valid anomaly detector representation.

Compilation flags:
`static`

Template:
`check_anomaly_detector(AnomalyDetector)`
Mode and number of proofs:
`check_anomaly_detector(@compound)` - `one_or_error`

Exceptions:
`AnomalyDetector` is a variable:
`instantiation_error`
`AnomalyDetector` is neither a variable nor a valid anomaly detector:
`domain_error(anomaly_detector,AnomalyDetector)`

---

valid_anomaly_detector/1

True when a learned anomaly detector term is structurally valid for the receiving implementation. Succeeds iff `check_anomaly_detector/1` succeeds without throwing an exception.

Compilation flags:
`static`

Template:
`valid_anomaly_detector(AnomalyDetector)`
Mode and number of proofs:
`valid_anomaly_detector(@compound)` - `zero_or_one`

---

diagnostics/2

Returns diagnostics and metadata associated with a learned anomaly detector in a representation-independent way.

Compilation flags:
```
static
```

Template:
```
diagnostics(AnomalyDetector,Diagnostics)
```
Mode and number of proofs:
```
diagnostics(+compound,-list(compound)) - one
```

---

diagnostic/2

Tests or enumerates individual diagnostics metadata terms for a learned anomaly detector.

Compilation flags:
```
static
```

Template:
```
diagnostic(AnomalyDetector,Diagnostic)
```
Mode and number of proofs:
```
diagnostic(+compound,?compound) - zero_or_more
```

---

anomaly_detector_options/2

Returns the effective training options recorded in a learned anomaly detector.

Compilation flags:
```
static
```

Template:
```
anomaly_detector_options(AnomalyDetector,Options)
```
Mode and number of proofs:
```
anomaly_detector_options(+compound,-list(compound)) - one
```

---

predict/3

Predicts whether a new instance is `normal` or `anomaly` using the learned detector. The instance is a list of `Attribute-Value` pairs.

Compilation flags:

```
static
```

Template:

```
predict(AnomalyDetector,Instance,Prediction)
```

Mode and number of proofs:

```
predict(+compound,+list,-atom) - one
```

---

predict/4

Predicts whether a new instance is `normal` or `anomaly` using the learned detector and the specified options. The instance is a list of `Attribute-Value` pairs.

Compilation flags:

```
static
```

Template:

```
predict(AnomalyDetector,Instance,Prediction,Options)
```

Mode and number of proofs:

```
predict(+compound,+list,-atom,+list(compound)) - one
```

---

score/3

Computes an anomaly score in the interval `[0.0, 1.0]` for a new instance. Larger scores indicate more anomalous instances.

Compilation flags:

```
static
```

Template:

```
score(AnomalyDetector,Instance,Score)
```

Mode and number of proofs:

```
score(+compound,+list,-float) - one
```

---

score_all/3

Computes anomaly scores for all examples in a dataset and returns a list of `Id-Class-Score` triples sorted by descending score.

Compilation flags:
`static`

Template:
`score_all(Dataset,AnomalyDetector,Scores)`
Mode and number of proofs:
`score_all(+object_identifier,+compound,-list)` - `one`

---

export_to_clauses/4

Converts an anomaly detector into a list of predicate clauses. `Functor` is the functor for the generated predicate clauses. When exporting a serialized detector term, a noun such as `detector` or `model` is usually clearer than a verb such as `detect`.

Compilation flags:
`static`

Template:
`export_to_clauses(Dataset,AnomalyDetector,Functor,Clauses)`
Mode and number of proofs:
`export_to_clauses(+object_identifier,+compound,+callable,-list(clause))` - `one`

---

export_to_file/4

Exports an anomaly detector to a file. `Functor` is the functor for the generated predicate clauses. When exporting a serialized detector term, a noun such as `detector` or `model` is usually clearer than a verb such as `detect`.

Compilation flags:
`static`

Template:
`export_to_file(Dataset,AnomalyDetector,Functor,File)`
Mode and number of proofs:
`export_to_file(+object_identifier,+compound,+callable,+atom)` - `one`

`print_anomaly_detector/1`

Prints an anomaly detector to the current output stream in a human-readable format.

Compilation flags:
`static`

Template:
`print_anomaly_detector(AnomalyDetector)`
Mode and number of proofs:
`print_anomaly_detector(+compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

anomaly_dataset_protocol, *anomaly_detector_common*, *isolation_forest_anomaly_detector*, *knn_distance_anomaly_detector*, *lof_anomaly_detector*

1.5 ant_colony

object

1.5.1 `ant_colony(Problem)`

- `Problem` - Problem object implementing `ant_colony_problem_protocol`.

Ant colony optimization algorithm using the Xoshiro128++ random number generator via `fast_random(xoshiro128pp)`.

Availability:
`logtalk_load(ant_colony(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Extends:

`public` *ant_colony(Problem,fast_random(xoshiro128pp))*

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ant_colony(Problem,Random), *ant_colony_problem_protocol*

object

1.5.2 `ant_colony(Problem,Random)`

- `Problem` - Problem object implementing `ant_colony_problem_protocol`.
- `Random` - Random number generator object providing `between/3`, `random/1`, and `randomize/1` predicates.

Ant Colony Optimization (Ant System) metaheuristic. Parameterized by a problem object implementing the `ant_colony_problem_protocol` protocol and by a random number generator algorithm for the `fast_random` library. The algorithm minimizes the solution cost defined by the problem. Classic Ant System pheromone update, optional MAX-MIN pheromone bounds, candidate construction by probabilistic state transition, optional elitist reinforcement, progress reporting, and seed control are supported; suitable defaults are used otherwise.

Availability:

`logtalk_load(ant_colony(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*

Uses:
list
type

Remarks:

- Algorithm: Ant Colony Optimization is a constructive metaheuristic inspired by the foraging behaviour of real ants. Artificial ants build solutions component by component on a construction graph, guided by pheromone trails and heuristic information. After each iteration the pheromone is evaporated and reinforced according to the quality of the constructed solutions.
- Pheromone model: Pheromone is stored on directed edges (From-To). Undirected problems should return symmetric heuristic values; the library treats edges as directed for generality.
- State transition: The probability of choosing next node `j` from `i` is proportional to `Tau_ij^Alpha * Eta_ij^Beta`. Roulette-wheel selection is used among the yet-unvisited nodes.
- Pheromone update: Classic Ant System: every edge evaporates by factor `(1-Rho)`, then each ant deposits `Q / Cost` on the edges of its tour. When `elite(E)` is greater than zero the global-best tour receives an extra `E * Q / BestCost` deposit. After each update every trail is clamped to the interval `[tau_min, tau_max]` (MAX-MIN style bounds).
- Candidate generation: Each ant constructs a complete tour by starting at a random node and repeatedly selecting the next unvisited node until the tour is closed.
- Progress reporting: If the problem object defines `progress/5`, it is called periodically with the current iteration, best cost, iteration-best cost, a placeholder acceptance rate, and improvement rate. The reporting interval is controlled by the `updates(N)` option. A final report is always produced when the loop terminates.
- Best solution tracking: The algorithm tracks the best solution found across all iterations.
- Seed control: The `seed(S)` option initializes the random number generator for reproducible runs.

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *run/2*
  - *run/3*
  - *run/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

run/2

Runs the ant colony algorithm using default options and returns the best solution found and its cost.

Compilation flags:
`static`

Template:
`run(BestSolution,BestCost)`
Mode and number of proofs:
`run(-list,-number)` - `one`

---

run/3

Runs the ant colony algorithm using the given options and returns the best solution found and its cost.

Compilation flags:
`static`

Template:
`run(BestSolution,BestCost,Options)`
Mode and number of proofs:
`run(-list,-number,+list(compound))` - `one`

Remarks:

- `max_iterations(N)` option: Maximum number of iterations (default: `100`).
- `ants(N)` option: Number of ants (solutions constructed) per iteration (default: `10`).
- `alpha(A)` option: Pheromone importance exponent (default: `1.0`).
- `beta(B)` option: Heuristic importance exponent (default: `2.0`).
- `rho(R)` option: Evaporation rate in `(0,1]` (default: `0.5`).
- `q(Q)` option: Pheromone deposit constant (default: `100.0`).
- `elite(E)` option: Elitist weight: extra deposit factor for the global-best tour (default: `0`).
- `tau0(T)` option: Initial pheromone level on every edge (default: `1.0`). Clamped into `[tau_min, tau_max]` at initialization.
- `tau_min(T)` option: Lower bound on pheromone trails; must be strictly positive (default: `1.0e-12`).
- `tau_max(T)` option: Upper bound on pheromone trails; must be strictly positive and at least `tau_min` (default: `1.0e300`).
- `updates(N)` option: Number of progress reports during the run. Set to `0` to disable (default: `0`).
- `seed(S)` option: Positive integer seed for the random number generator, enabling reproducible runs (default: none).

`run/4`

Runs the ant colony algorithm using the given options, returns the best solution found and its cost, and returns run statistics.

Compilation flags:
`static`

Template:
`run(BestSolution,BestCost,Statistics,Options)`
Mode and number of proofs:
`run(-list,-number,-list(compound),+list(compound))` - `one`

Remarks:

- Statistics list: A list of `Key(Value)` pairs: `iterations(N)` is the number of iterations executed, `solutions(S)` is the total number of solutions constructed, `improvements(I)` is the number of times the global best was improved, and `final_best_cost(C)` is the best cost found.

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ant_colony(Problem), *ant_colony_problem_protocol*

protocol

1.5.3 `ant_colony_problem_protocol`

Protocol for ant colony optimization problem definitions. A problem object must define the required predicates for constructive solution building on a construction graph (typically a complete graph over a set of nodes, as in the TSP). Optional predicates allow custom stopping and progress reporting.

Availability:
`logtalk_load(ant_colony(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-16

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *nodes/1*
  - *heuristic/3*
  - *solution_cost/2*
  - *stop_condition/3*
  - *progress/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

nodes/1

Returns the ordered list of nodes that form the construction graph. Solutions are permutations (or paths) over these nodes.

Compilation flags:
`static`

Template:
`nodes(Nodes)`
Mode and number of proofs:
`nodes(-list)` - `one`

---

heuristic/3

Returns the heuristic desirability `Eta` of moving from `From` to `To`. Typically the reciprocal of a distance or cost. Must be strictly positive for every pair that can appear in a solution.

Compilation flags:
`static`

Template:
`heuristic(From,To,Eta)`
Mode and number of proofs:
`heuristic(+nonvar,+nonvar,-number)` - `one`

---

solution_cost/2

Computes the cost (energy) of a complete solution (tour/path). The algorithm minimizes this value.

Compilation flags:

```
static
```

Template:

```
solution_cost(Solution,Cost)
```

Mode and number of proofs:

```
solution_cost(+list,-number) - one
```

---

stop_condition/3

True when the search should stop given the current iteration, best cost found so far, and the cost of the iteration-best solution. Optional. When not defined by the problem, the search runs until the maximum number of iterations is reached.

Compilation flags:

```
static
```

Template:

```
stop_condition(Iteration,BestCost,IterationBestCost)
```

Mode and number of proofs:

```
stop_condition(+non_negative_integer,+number,+number) - zero_or_one
```

---

progress/5

Called periodically to report optimization progress. Optional. When not defined by the problem, progress reporting is skipped. The rates are values between 0.0 and 1.0.

Compilation flags:

```
static
```

Template:

```
progress(Iteration,BestCost,IterationBestCost,AcceptanceRate,ImprovementRate)
```

Mode and number of proofs:

```
progress(+non_negative_integer,+number,+number,+number,+number) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

ant_colony(Problem)

1.6 application

category

1.6.1 `application_common`

Application metadata and provenance predicates.

Availability:
`logtalk_load(application(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-03-25

Compilation flags:
`static`

Implements:
`public` *application_protocol*
Uses:
os

Remarks:
(none)

Inherited public predicates:
built_date/1 *creators/1* *description/1* *distribution/1* *external_reference/2* *git_object_identifier/1* *homepage/1* *license/1* *loader_file/1* *name/1* *originator/1* *package/1* *property/1* *release_date/1* *repository/1* *repository_branch/1* *repository_commit/1* *repository_commit_abbreviated/1* *repository_commit_author/1* *repository_commit_date/1* *repository_commit_message/1* *software_heritage_identifier/1* *supplier/1* *valid_until_date/1* *version/1*

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.6.2 application_protocol

Application metadata predicates, including optional source provenance facts.

Availability:
`logtalk_load(application(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-03-25

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- Explicit metadata: All predicates describe metadata declared by the application. The protocol does not require or imply reflection over the current and transient state of some application repository.
- Release metadata: Predicates such as `name/1`, `version/1`, `description/1`, `license/1`, `distribution/1`, `package/1`, `release_date/1`, and `valid_until_date/1` are intended for release-oriented metadata.
- Source provenance metadata: Predicates such as `repository/1`, `repository_branch/1`, `repository_commit/1`, `repository_commit_date/1`, `repository_commit_author/1`, `repository_commit_message/1`, `git_object_identifier/1`, and `software_heritage_identifier/1` are optional explicit provenance facts about the source used for the application and may or may not correspond to the release artifact identity.

Inherited public predicates:

(none)

- *Public predicates*
  - *name/1*
  - *version/1*
  - *description/1*
  - *license/1*
  - *homepage/1*
  - *distribution/1*
  - *package/1*
  - *loader_file/1*
  - *creators/1*
  - *supplier/1*
  - *originator/1*
  - *built_date/1*
  - *release_date/1*
  - *valid_until_date/1*
  - *external_reference/2*
  - *repository/1*
  - *repository_branch/1*
  - *repository_commit/1*
  - *repository_commit_abbreviated/1*
  - *repository_commit_date/1*
  - *repository_commit_author/1*
  - *repository_commit_message/1*
  - *git_object_identifier/1*
  - *software_heritage_identifier/1*
  - *property/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

name/1

Application name.

Compilation flags:
`static`

Template:
`name(Name)`

Mode and number of proofs:
`name(?atom)` - `zero_or_one`

version/1

Application version.

Compilation flags:
`static`

Template:
`version(Version)`
Mode and number of proofs:
`version(?atom)` - `zero_or_one`

---

description/1

Application short description.

Compilation flags:
`static`

Template:
`description(Description)`
Mode and number of proofs:
`description(?atom)` - `zero_or_one`

---

license/1

Application license.

Compilation flags:
`static`

Template:
`license(License)`
Mode and number of proofs:
`license(?atom)` - `zero_or_one`

---

homepage/1

Application homepage URL.

Compilation flags:
`static`

Template:
`homepage(URL)`
Mode and number of proofs:
`homepage(?atom)` - `zero_or_one`

---

distribution/1

Application distribution or download location.

Compilation flags:
`static`

Template:
`distribution(URL)`
Mode and number of proofs:
`distribution(?atom)` - `zero_or_one`

---

package/1

Application package identifier as a PURL.

Compilation flags:
`static`

Template:
`package(PURL)`
Mode and number of proofs:
`package(?atom)` - `zero_or_one`

---

loader_file/1

Application main loader file absolute path.

Compilation flags:
`static`

Template:
`loader_file(File)`
Mode and number of proofs:
`loader_file(?atom)` - `zero_or_one`

---

creators/1

Application creators, authors, or other credited producers of the application or its release metadata.

Compilation flags:
`static`

Template:
`creators(Creators)`
Mode and number of proofs:
`creators(?list(atom))` - `zero_or_one`

---

supplier/1

Application supplier.

Compilation flags:
`static`

Template:
`supplier(Supplier)`
Mode and number of proofs:
`supplier(?atom)` - `zero_or_one`

---

originator/1

Original source of the application software when distinct from its creators.

Compilation flags:
`static`

Template:
`originator(Originator)`
Mode and number of proofs:
`originator(?atom)` - `zero_or_one`

---

built_date/1

Application build date.

Compilation flags:
`static`

Template:
`built_date(Date)`
Mode and number of proofs:
`built_date(?atom)` - `zero_or_one`

---

release_date/1

Application release date.

Compilation flags:
`static`

Template:
`release_date(Date)`
Mode and number of proofs:
`release_date(?atom)` - `zero_or_one`

---

valid_until_date/1

Application validity limit date.

Compilation flags:
`static`

Template:
`valid_until_date(Date)`
Mode and number of proofs:
`valid_until_date(?atom)` - `zero_or_one`

---

external_reference/2

Application explicit external references using the same vocabulary as the corresponding first-class metadata predicates. The second argument can be a URL or a non-URL identifier depending on the reference type.

Compilation flags:
`static`

Template:
`external_reference(Type,Locator)`
Mode and number of proofs:
`external_reference(?atom,?atom)` - `zero_or_more`

---

repository/1

Application source provenance repository metadata.

Compilation flags:
`static`

Template:
`repository(URL)`
Mode and number of proofs:
`repository(?atom)` - `zero_or_one`

---

repository_branch/1

Application source provenance git branch metadata.

Compilation flags:
`static`

Template:
`repository_branch(Branch)`
Mode and number of proofs:
`repository_branch(?atom)` - `zero_or_one`

---

repository_commit/1

Application source provenance git commit metadata.

Compilation flags:
`static`

Template:
`repository_commit(Hash)`
Mode and number of proofs:
`repository_commit(?atom)` - `zero_or_one`

---

repository_commit_abbreviated/1

Application abbreviated source provenance git commit metadata.

Compilation flags:
`static`

Template:
`repository_commit_abbreviated(Hash)`
Mode and number of proofs:
`repository_commit_abbreviated(?atom)` - `zero_or_one`

---

repository_commit_date/1

Application source provenance git commit date metadata.

Compilation flags:
`static`

Template:
`repository_commit_date(Date)`
Mode and number of proofs:
`repository_commit_date(?atom)` - `zero_or_one`

---

repository_commit_author/1

Application source provenance git commit author metadata.

Compilation flags:
`static`

Template:
`repository_commit_author(Author)`
Mode and number of proofs:
`repository_commit_author(?atom)` - `zero_or_one`

---

repository_commit_message/1

Application source provenance git commit message metadata.

Compilation flags:
`static`

Template:
`repository_commit_message(Message)`
Mode and number of proofs:
`repository_commit_message(?atom)` - `zero_or_one`

---

git_object_identifier/1

Application source provenance git object identifier metadata as a gitoid.

Compilation flags:
`static`

Template:
`git_object_identifier(GITOID)`
Mode and number of proofs:
`git_object_identifier(?atom)` - `zero_or_one`

---

software_heritage_identifier/1

Application source provenance Software Heritage identifier metadata as an SWHID.

Compilation flags:
`static`

Template:
`software_heritage_identifier(SWHID)`
Mode and number of proofs:
`software_heritage_identifier(?atom)` - `zero_or_one`

---

property/1

Enumerates declared application metadata and optional source provenance as individual property terms.

Compilation flags:
`static`

Template:
`property(Property)`
Mode and number of proofs:
`property(?compound)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.7 apriori_pattern_miner

object

1.7.1 `apriori_pattern_miner`

Apriori frequent itemset miner for transaction datasets using deterministic level-wise candidate generation and anti-monotone pruning.

Availability:
`logtalk_load(apriori_pattern_miner(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *frequent_pattern_mining_common*
Uses:
avltree
combinations
fnv1a_32
format
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_pattern_miner/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *mine/2* *mine/3* *option/2* *option/3* *pattern_miner_options/2* *print_pattern_miner/1* *valid_option/1* *valid_options/1* *valid_pattern_miner/1*

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pattern_miner_protocol, *transaction_dataset_protocol*, *combinations*

1.8 arbitrary

category

1.8.1 `arbitrary`

Adds predicates for generating and shrinking random values for selected types to the library `type` object. User extensible.

Availability:
`logtalk_load(arbitrary(loader))`

Author: Paulo Moura
Version: 2:39:1
Date: 2026-11-09

Compilation flags:
`static`

Complements:
type
Uses:
fast_random
integer
list
type

Remarks:

- Logtalk specific types: `entity`, `object`, `protocol`, `category`, `entity_identifier`, `object_identifier`, `protocol_identifier`, `category_identifier`, `event`, `predicate`.
- Prolog module related types (when the backend compiler supports modules): `module`, `module_identifier`, `qualified_callable`.
- Prolog base types: `term`, `var`, `nonvar`, `atomic`, `atom`, `number`, `integer`, `float`, `compound`, `callable`, `ground`.
- Atom derived types: `non_quoted_atom`, `non_empty_atom`, `non_empty_atom(CharSet)`, `boolean`, `character`, `in_character`, `char`, `operator_specifier`, `hex_char`.
- Atom derived parametric types: `atom(CharSet)`, `atom(CharSet,Length)`, `non_empty_atom(CharSet)`, `character(CharSet)`, `in_character(CharSet)`, `char(CharSet)`.
- Number derived types: `positive_number`, `negative_number`, `non_positive_number`, `non_negative_number`.
- Float derived types: `positive_float`, `negative_float`, `non_positive_float`, `non_negative_float`, `probability`, `open_probability`.
- Integer derived types: `positive_integer`, `negative_integer`, `non_positive_integer`, `non_negative_integer`, `byte`, `in_byte`, `character_code`, `in_character_code`, `code`, `operator_priority`, `hex_code`.
- Integer derived parametric types: `character_code(CharSet)`, `in_character_code(CharSet)`, `code(CharSet)`.
- List types (compound derived types): `list`, `non_empty_list`, `partial_list`, `list_or_partial_list`, `list(Type)`, `list(Type,Length)`, `list(Type,Min,Max)`, `list(Type,Length,Min,Max)`, `non_empty_list(Type)`, `codes`, `chars`.
- Difference list types (compound derived types): `difference_list`, `difference_list(Type)`.
- List and difference list types length: The types that do not take a fixed length generate lists with a length in the `[0,MaxSize]` interval (`[1,MaxSize]` for non-empty list types).
- Predicate and non-terminal indicator types arity: These types generate indicators with an arity in the `[0,MaxSize]` interval.
- Other compound derived types: `compound(Name,Types)`, `predicate_indicator`, `non_terminal_indicator`, `predicate_or_non_terminal_indicator`, `clause`, `grammar_rule`, `pair`, `pair(KeyType,ValueType)`.
- Other types: `text`, `text(CharSet)`, `Object::Closure`, `between(Type,Lower,Upper)`, `property(Type,LambdaExpression)`, `one_of(Type,Set)`, `var_or(Type)`, `ground(Type)`, `types(Types)`, `types_frequency(Pairs)`, `transform(Type,Closure)`, `constrain(Type,Closure)`.
- Types `text` and `text(CharSet)` notes: Generate random text represented using either atoms, character lists, or character code lists.
- Type `Object::Closure` notes: Allows calling public object predicates as generators and shrinkers. The `Closure` closure is extended with either a single argument, the generated arbitrary value, or with two arguments, when shrinking a value.
- Type `compound(Name,Types)` notes: Generate a random compound term with the given name with a random argument for each type.
- Type `types_frequency(Pairs)` notes: Generate a random term for one of the types in a list of `Type-Frequency` pairs. The type is randomly selected taking into account the types frequency.
- Type `transform(Type,Closure)` notes: Generate a random term by transforming the term generated for the given type using the given closure.
- Type `constrain(Type,Closure)` notes: Generate a random term for the given type that satisfy the given closure.
- Registering new types: Add clauses for the `arbitrary/1-2` multifile predicates and optionally for the `shrinker/1` and `shrink/3` multifile predicates. The clauses must have a bound first argument to avoid introducing spurious choice-points.
- Shrinking values: The `shrink/3` should either succeed or fail but never throw an exception.
- Character sets: `ascii_identifier`, `ascii_printable`, `ascii_full`, `hexadecimal`, `byte`, `unicode_bmp`, `unicode_full`.
- Default character sets: The default character set when using a parameterizable type that takes a character set parameter depends on the type.

- Default character sets: Entity, predicate, and non-terminal identifier types plus compound and callable types default to an `ascii_identifier` functor. Character and character code types default to `ascii_full`. Other types default to `ascii_printable`.
- Caveats: The type argument (and any type parameterization) to the predicates is not type-checked (or checked for consistency) for performance reasons.
- Unicode limitations: Currently, correct character/code generation is only ensured for SWI-Prolog and XVM as other backends do not provide support for querying a Unicode code point category.

Inherited public predicates:

(none)

- *Public predicates*
  - *arbitrary/1*
  - *arbitrary/2*
  - *shrinker/1*
  - *shrink/3*
  - *shrink_sequence/3*
  - *edge_case/2*
  - *get_seed/1*
  - *set_seed/1*
  - *max_size/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

arbitrary/1

Table of defined types for which an arbitrary value can be generated. A new type can be registered by defining a clause for this predicate and adding a clause for the `arbitrary/2` multifile predicate.

Compilation flags:

```
static, multifile
```

Template:

```
arbitrary(Type)
```

Mode and number of proofs:

```
arbitrary(?callable) - zero_or_more
```

arbitrary/2

Generates an arbitrary term of the specified type. Fails if the type is not supported. A new generator can be defined by adding a clause for this predicate and registering it via the `arbitrary/1` predicate.

Compilation flags:
`static, multifile`

Template:
`arbitrary(Type,Term)`
Meta-predicate template:
`arbitrary(::,*)`
Mode and number of proofs:
`arbitrary(@callable,-term)` - `zero_or_one`

---

shrinker/1

Table of defined types for which a shrinker is provided. A new shrinker can be registered by defining a clause for this predicate and adding a definition for the `shrink/3` multifile predicate.

Compilation flags:
`static, multifile`

Template:
`shrinker(Type)`
Mode and number of proofs:
`shrinker(?callable)` - `zero_or_more`

---

shrink/3

Shrinks a value to a smaller value if possible. Must generate a finite number of solutions. Fails if the type is not supported. A new shrinker can be defined by adding a clause for this predicate and registering it via the `shrinker/1` predicate.

Compilation flags:
`static, multifile`

Template:
`shrink(Type,Large,Small)`
Mode and number of proofs:
`shrink(@callable,@term,-term)` - `zero_or_more`

---

shrink_sequence/3

Shrinks a value repeatedly until shrinking is no longer possible returning the sequence of values (ordered from larger to smaller value). Fails if the type is not supported.

Compilation flags:

```
static
```

Template:

```
shrink_sequence(Type,Value,Sequence)
```

Mode and number of proofs:

```
shrink_sequence(@callable,@term,-list(term)) - zero_or_one
```

---

edge_case/2

Table of type edge cases. Fails if the given type have no defined edge cases. New edge cases for existing or new types can be added by defining a clause for this multifile predicate.

Compilation flags:

```
static, multifile
```

Template:

```
edge_case(Type,Term)
```

Mode and number of proofs:

```
edge_case(?callable,?term) - zero_or_more
```

---

get_seed/1

Gets the current random generator seed. Seed should be regarded as an opaque ground term.

Compilation flags:

```
static
```

Template:

```
get_seed(Seed)
```

Mode and number of proofs:

```
get_seed(-ground) - one
```

---

set_seed/1

Sets the random generator seed to a given value returned by calling the `get_seed/1` predicate.

Compilation flags:
`static`

Template:
`set_seed(Seed)`
Mode and number of proofs:
`set_seed(+ground)` - `one`

---

max_size/1

User defined maximum size for types where its meaningful and implicit. When not defined, defaults to 42. When multiple definitions exist, the first valid one found is used.

Compilation flags:
`static, multifile`

Template:
`max_size(Size)`
Mode and number of proofs:
`max_size(?positive_integer)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

type

1.9 arrangements

object

1.9.1 `arrangements`

Implementation of arrangements operations with repetition over lists.

Availability:
`logtalk_load(arrangements(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-05-21

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *arrangements_protocol*
Uses:
fast_random(Algorithm)
list

Remarks:
(none)

Inherited public predicates:
arrangement/3 *arrangement/4* *arrangement_index/4* *arrangement_index/5* *arrangements/3* *arrangements/4* *count_arrangements/3* *count_distinct_arrangements/3* *distinct_arrangement/3* *distinct_arrangement/4* *distinct_arrangement_index/4* *distinct_arrangements/3* *distinct_arrangements/4* *next_arrangement/3* *nth_arrangement/4* *nth_arrangement/5* *nth_distinct_arrangement/4* *previous_arrangement/3* *random_arrangement/3* *random_distinct_arrangement/3* *sample_arrangements/4* *sample_distinct_arrangements/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.9.2 `arrangements_protocol`

Protocol for arrangements operations over lists.

Availability:
`logtalk_load(arrangements(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-12

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *arrangements/3*
  - *arrangement/3*
  - *arrangements/4*
  - *arrangement/4*
  - *distinct_arrangements/3*
  - *distinct_arrangement/3*
  - *distinct_arrangements/4*
  - *distinct_arrangement/4*
  - *nth_arrangement/4*
  - *nth_arrangement/5*
  - *arrangement_index/4*

  - *arrangement_index/5*
  - *count_arrangements/3*
  - *count_distinct_arrangements/3*
  - *nth_distinct_arrangement/4*
  - *distinct_arrangement_index/4*
  - *random_arrangement/3*
  - *sample_arrangements/4*
  - *random_distinct_arrangement/3*
  - *sample_distinct_arrangements/4*
  - *next_arrangement/3*
  - *previous_arrangement/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

arrangements/3

Generates all K-arrangements (ordered K-element selections with replacement) of a list using default order.

Compilation flags:

```
static
```

Template:

```
arrangements(K,List,Arrangements)
```

Mode and number of proofs:

```
arrangements(+integer,+list,-list) - one
```

---

arrangement/3

True iff the third argument is a K-arrangement with replacement of a list using default order.

Compilation flags:

```
static
```

Template:

```
arrangement(K,List,Arrangement)
```

Mode and number of proofs:

```
arrangement(+integer,+list,-list) - one_or_more
```

---

arrangements/4

Generates all K-arrangements with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`arrangements(K,List,Order,Arrangements)`
Mode and number of proofs:
`arrangements(+integer,+list,+atom,-list)` - `one`

---

arrangement/4

True iff the fourth argument is a K-arrangement with replacement with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`arrangement(K,List,Order,Arrangement)`
Mode and number of proofs:
`arrangement(+integer,+list,+atom,-list)` - `one_or_more`

---

distinct_arrangements/3

Generates all distinct K-arrangements (deduplicating repeated values in the input list) using default order.

Compilation flags:
`static`

Template:
`distinct_arrangements(K,List,Arrangements)`
Mode and number of proofs:
`distinct_arrangements(+integer,+list,-list)` - `one`

---

distinct_arrangement/3

True iff the third argument is a distinct K-arrangement with replacement of a list using default order.

Compilation flags:
`static`

Template:
`distinct_arrangement(K,List,Arrangement)`
Mode and number of proofs:
`distinct_arrangement(+integer,+list,-list)` - `one_or_more`

---

distinct_arrangements/4

Generates all distinct K-arrangements with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_arrangements(K,List,Order,Arrangements)`
Mode and number of proofs:
`distinct_arrangements(+integer,+list,+atom,-list)` - `one`

---

distinct_arrangement/4

True iff the fourth argument is a distinct K-arrangement with replacement with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_arrangement(K,List,Order,Arrangement)`
Mode and number of proofs:
`distinct_arrangement(+integer,+list,+atom,-list)` - `one_or_more`

---

nth_arrangement/4

Returns the K-arrangement at a given zero-based index using default order.

Compilation flags:
`static`

Template:
`nth_arrangement(K,List,Index,Arrangement)`
Mode and number of proofs:
`nth_arrangement(+integer,+list,+integer,-list)` - `zero_or_one`

---

nth_arrangement/5

Returns the K-arrangement at a given zero-based index in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`nth_arrangement(K,List,Order,Index,Arrangement)`
Mode and number of proofs:
`nth_arrangement(+integer,+list,+atom,+integer,-list)` - `zero_or_one`

---

arrangement_index/4

Returns the zero-based index of a K-arrangement using default order.

Compilation flags:
`static`

Template:
`arrangement_index(K,List,Arrangement,Index)`
Mode and number of proofs:
`arrangement_index(+integer,+list,+list,-integer)` - `zero_or_one`

---

arrangement_index/5

Returns the zero-based index of a K-arrangement in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`arrangement_index(K,List,Order,Arrangement,Index)`
Mode and number of proofs:
`arrangement_index(+integer,+list,+atom,+list,-integer)` - `zero_or_one`

---

count_arrangements/3

Counts the number of K-arrangements with replacement of a list.

Compilation flags:
`static`

Template:
`count_arrangements(K,List,Count)`
Mode and number of proofs:
`count_arrangements(+integer,+list,-integer)` - `one`

---

count_distinct_arrangements/3

Counts the number of distinct K-arrangements with replacement of a list (deduplicating repeated values in the input list).

Compilation flags:
`static`

Template:
`count_distinct_arrangements(K,List,Count)`
Mode and number of proofs:
`count_distinct_arrangements(+integer,+list,-integer)` - `one`

---

nth_distinct_arrangement/4

Returns the distinct K-arrangement at a given zero-based index in default generation order.

Compilation flags:
```
static
```

Template:
```
nth_distinct_arrangement(K,List,Index,Arrangement)
```
Mode and number of proofs:
```
nth_distinct_arrangement(+integer,+list,+integer,-list) - zero_or_one
```

---

distinct_arrangement_index/4

Returns the zero-based index of a distinct K-arrangement in default generation order.

Compilation flags:
```
static
```

Template:
```
distinct_arrangement_index(K,List,Arrangement,Index)
```
Mode and number of proofs:
```
distinct_arrangement_index(+integer,+list,+list,-integer) - zero_or_one
```

---

random_arrangement/3

Returns a random K-arrangement with replacement of a list.

Compilation flags:
```
static
```

Template:
```
random_arrangement(K,List,Arrangement)
```
Mode and number of proofs:
```
random_arrangement(+integer,+list,-list) - zero_or_one
```

---

sample_arrangements/4

Returns SampleCount random K-arrangements with replacement of a list, sampled with replacement.

Compilation flags:
```
static
```

Template:
```
sample_arrangements(K,List,SampleCount,Samples)
```
Mode and number of proofs:
```
sample_arrangements(+integer,+list,+integer,-list) - zero_or_one
```

---

random_distinct_arrangement/3

Returns a random distinct K-arrangement with replacement of a list (deduplicating repeated values in the input list).

Compilation flags:
```
static
```

Template:
```
random_distinct_arrangement(K,List,Arrangement)
```
Mode and number of proofs:
```
random_distinct_arrangement(+integer,+list,-list) - zero_or_one
```

---

sample_distinct_arrangements/4

Returns SampleCount random distinct K-arrangements with replacement of a list, sampled with replacement after deduplicating repeated values in the input list.

Compilation flags:
```
static
```

Template:
```
sample_distinct_arrangements(K,List,SampleCount,Samples)
```
Mode and number of proofs:
```
sample_distinct_arrangements(+integer,+list,+integer,-list) - zero_or_one
```

---

next_arrangement/3

Returns the next distinct arrangement value in lexicographic order induced by the first argument.

Compilation flags:
```
static
```

Template:
```
next_arrangement(List,Arrangement,Next)
```
Mode and number of proofs:
```
next_arrangement(+list,+list,-list) - zero_or_one
```

---

previous_arrangement/3

Returns the previous distinct arrangement value in lexicographic order induced by the first argument.

Compilation flags:
```
static
```

Template:
```
previous_arrangement(List,Arrangement,Previous)
```
Mode and number of proofs:
```
previous_arrangement(+list,+list,-list) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.10 asp

object

1.10.1 asp

Process-backed Clingo and Zetesis CLI solving, ASP source generation, and collected result inspection.

Availability:
`logtalk_load(asp(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-09

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *asp_protocol*
Imports:
`public` *options*
Uses:
asp_output
asp_process
asp_program
list
os
reader
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *model/2* *option/2* *option/3* *print_program/1* *print_result/1* *program/1* *program/2* *result_property/2* *solve/2* *solve/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.10.2 asp_expansion

Opt-in equivalent ASP directive and clause expansion for pure problem objects.

Availability:
`logtalk_load(asp(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-10

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
asp_program
list
type

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.10.3 asp_output

Whole-document Clingo and Zetesis JSON output decoding.

Availability:
`logtalk_load(asp(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-15

Compilation flags:
`static, context_switching_calls`

Uses:
asp_program
asp_symbols
json(ObjectRepresentation,PairRepresentation,StringRepresentation)
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *decode/8*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

decode/8

Decodes CLI JSON and process metadata into a collected `asp_result(Status, Models, Properties)` term.

Compilation flags:

`static`

Template:

`decode(Solver,Codes,RawStatus,NormalizedStatus,Diagnostics,SourceMap,Result,Options)`

Mode and number of proofs:

`decode(+atom,+list(code),+term,+compound,+list(code),+list(compound),-compound,+list(compound))` - `one_or_error`

Exceptions:

`Solver` is a variable:

`instantiation_error`

`Solver` is neither a variable nor an atom:

`type_error(atom,Solver)`

`Solver` is an atom but not a valid ASP solver:

`domain_error(asp_solver,Solver)`

`Codes` is a partial list or a list with an element `Element` which is a variable:

`instantiation_error`

`Codes` is neither a partial list nor a list of character codes:

`type_error(codes,Codes)`

`Codes` does not contain parsable JSON:

`domain_error(json,codes(Codes))`

A JSON field has an invalid type or value:

`domain_error(asp_output,Field-Value)`

A JSON field required by the selected solver output schema is missing:

`domain_error(asp_output,Field-missing)`

The JSON simultaneously reports a time limit and another interruption:

`domain_error(asp_output,interruption_flags)`

A shown value is not a supported symbol:

`domain_error(asp_symbol,Codes)`

A numeric symbol cannot be represented as an integer:

`representation_error(asp_integer)`

`RawStatus` is not a ground term:

`instantiation_error`

`NormalizedStatus`` is not a ground term:

`instantiation_error`

`NormalizedStatus` is fatal or inconsistent with the decoded solver result:

`domain_error(asp_process_status,NormalizedStatus)`

`Diagnostics` is a partial list or a list with an element `Element` which is a variable:

`instantiation_error`

`Diagnostics` is neither a partial list nor a list of character codes:

`type_error(codes,Diagnostics)`

`SourceMap` is a partial list or a list with an element `Element` which is a variable:

`instantiation_error`

`SourceMap` is neither a variable nor a list:
`type_error(list,SourceMap)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.10.4 `asp_problem_protocol`

Protocol for ordered, ground ASP statement providers.

Availability:
`logtalk_load(asp(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-09

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *asp_statement/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

asp_statement/2

Enumerates, by backtracking, canonical ASP statements in source order and their origins.

Compilation flags:
`static`

Template:
`asp_statement(Statement,Origin)`
Mode and number of proofs:
`asp_statement(-compound,-term)` - `zero_or_more`

Remarks:

- Origin: Either `source(File, Start, End)` or `unknown`.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.10.5 asp_process

Isolated operating-system shell transport for ASP solver CLIs.

Availability:
`logtalk_load(asp(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Uses:
list
os
process
reader
type
user
utf_8

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *quote/3*
  - *run/7*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

quote/3

Quotes an atom as one POSIX shell or Windows PowerShell literal, without interpolation.

Compilation flags:
`static`

Template:
`quote(Platform,Argument,Quoted)`
Mode and number of proofs:
`quote(+atom,+atom,-atom)` - `one_or_error`

Exceptions:
`Platform` is a variable:
`instantiation_error`
`Platform` is neither a variable nor an atom:
`type_error(atom,Platform)`
`Platform` is an atom but neither `posix` nor `windows`:
`domain_error(asp_shell_platform,Platform)`
`Argument` is a variable:
`instantiation_error`

`Argument` is neither a variable nor an atom:
`type_error(atom,Argument)`
`Argument` is an atom but contains a null character:
`domain_error(asp_shell_argument,Argument)`

`run/7`

Runs the CLI using an owned directory and separate UTF-8 output files, returning only after cleanup.

Compilation flags:
`static`

Template:
`run(Executable,Arguments,Source,Codes,Diagnostics,RawStatus,NormalizedStatus)`
Mode and number of proofs:
`run(+atom,+list(atom),+compound,-list(code),-list(code),-term,-compound)` - `one_or_error`

Exceptions:
`Executable` is a variable:
`instantiation_error`
`Executable` is neither a variable nor an atom:
`type_error(atom,Executable)`
`Arguments` is a partial list or a list with an element `Argument` which is a variable:
`instantiation_error`
`Arguments` is neither a variable nor a list:
`type_error(list(atom),Arguments)`
An element `Argument` of the list `Arguments` is neither a variable nor an atom:
`type_error(atom,Argument)`
`Executable` or an element `Argument` of the list `Arguments` contains a null character:
`domain_error(asp_shell_argument,Argument)`
`Source` is not a ground term:
`instantiation_error`
`Source` is neither `file(Path)` nor `codes(Codes)`:
`domain_error(asp_process_source,Source)`
A launcher dependency does not exist:
`existence_error(command,Command)`
A source or capture file does not exist:
`existence_error(source_sink,Path)`
A source or capture file cannot be opened:
`permission_error(open,source_sink,Path)`
An owned directory cannot be found during cleanup:
`existence_error(directory,Directory)`
An owned file or directory cannot be removed:
`permission_error(delete,source_sink,Path)`
A character sequence is not valid UTF-8:
`representation_error(utf_8)`

A launcher operation fails:

```
asp_transport(Stage,Status,Details)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.10.6 asp_program

Canonical ASP statement validation and source generation.

Availability:

```
logtalk_load(asp(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-15

Compilation flags:

```
static, context_switching_calls
```

Uses:

list
type

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *serialize/2*
  - *serialize/4*
  - *symbol_codes/2*
  - *valid_identifier/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

serialize/2

Serializes ground canonical ASP statements to character codes.

Compilation flags:
`static`

Template:
`serialize(Statements,Codes)`
Mode and number of proofs:
`serialize(+list,-list(integer))` - `one_or_error`

Exceptions:
`Statements` is a partial list or a list with an element `Statement` which is a variable:
`instantiation_error`
`Statements` is neither a partial list nor a list:
`type_error(list,Statements)`
An element `Statement` of the list `Statements` is not a valid statement:
`domain_error(asp_statement,Statement)`

---

serialize/4

Serializes statements with origins and returns `line(Start, End, Origin)` source-map entries.

Compilation flags:
`static`

Template:
`serialize(Statements,Origins,Codes,SourceMap)`
Mode and number of proofs:
`serialize(+list,+list,-list(code),-list(compound))` - `one_or_error`

Exceptions:
`Statements` is a partial list or a list with an element `Statement` which is a variable:
`instantiation_error`
`Statements` is neither a partial list nor a list:
`type_error(list,Statements)`
An element `Statement` of the list `Statements` is not a valid statement:
`domain_error(asp_statement,Statement)`
`Origins` is a partial list or a list with an element `Origin` which is a variable:
`instantiation_error`
`Origins` is neither a variable nor a list:
`type_error(list,Origins)`
`Origins` has an element that is neither `unknown` nor a valid `source(File, Start, End)` term, or its length differs from `Statements`:
`domain_error(asp_origins,Origins)`

symbol_codes/2

Serializes a canonical ground symbol to ASP character codes.

Compilation flags:
`static`

Template:
`symbol_codes(Symbol,Codes)`
Mode and number of proofs:
`symbol_codes(+term,-list(integer))` - `one_or_error`

Exceptions:
`Symbol` is not a ground term:
`instantiation_error`
`Symbol` is ground but not a supported symbol:
`domain_error(asp_symbol,Symbol)`

valid_identifier/1

True if the argument is a supported ASP function or predicate identifier.

Compilation flags:
`static`

Template:
`valid_identifier(Name)`
Mode and number of proofs:
`valid_identifier(@term)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.10.7 asp_protocol

Protocol for process-backed ASP solving, source generation, and result inspection.

Availability:
`logtalk_load(asp(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *solve/2*
  - *solve/3*
  - *program/1*
  - *program/2*
  - *print_program/1*
  - *model/2*
  - *result_property/2*
  - *print_result/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

solve/2

Solves a source with default options using the external Clingo CLI and returns a collected result after cleanup. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`.

Compilation flags:

`static`

Template:

`solve(Source,Result)`

Mode and number of proofs:

`solve(+compound,-compound)` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `asp_problem_protocol`:

`domain_error(asp_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(asp_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(asp_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

A launcher argument contains a null character:

`domain_error(asp_shell_argument,Argument)`

The solver executable or a shell dependency does not exist:

`existence_error(command,Command)`

An owned directory cannot be found during cleanup:

`existence_error(directory,Directory)`

An owned file or directory cannot be removed:

`permission_error(delete,source_sink,Path)`

A character sequence is not valid UTF-8:

`representation_error(utf_8)`

A launcher operation fails:

`asp_transport(Stage,Status,Details)`

Solver output is invalid or inconsistent with its exit status:
`asp_solver_error(Formal,Status,Diagnostics,SourceMap)`

`solve/3`

Solves a source using solver selection, executable, enumeration, optimization, resource, and statistics options. Repeated options use the first occurrence. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`.

Compilation flags:
`static`

Template:
`solve(Source,Result,Options)`

Mode and number of proofs:
`solve(+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Source` is not a ground term:
`instantiation_error`
`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `asp_problem_protocol`:
`domain_error(asp_source,Source)`
`Statements` is ground but not a list:
`type_error(list,Statements)`
A statement enumerated by the provider object is not a valid statement:
`domain_error(asp_statement,Statement)`
An origin enumerated by the provider object is invalid, or its origins and statements differ in length:
`domain_error(asp_origins,Origins)`
`Codes` is ground but not a list of character codes:
`type_error(codes,Codes)`
An element `Code` of the list `Codes` is ground but not an integer:
`type_error(integer,Code)`
An element `Code` of the list `Codes` is an integer but not a valid character code:
`domain_error(character_code,Code)`
`Path` is ground but not an atom:
`type_error(atom,Path)`
`Path` does not exist:
`existence_error(source_sink,Path)`
`Path` cannot be opened for reading:
`permission_error(open,source_sink,Path)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

An effective option is unsupported by the selected solver:

`domain_error(asp_solver_option,Solver-Option)`

A launcher argument contains a null character:

`domain_error(asp_shell_argument,Argument)`

The solver executable or a shell dependency does not exist:

`existence_error(command,Command)`

An owned directory cannot be found during cleanup:

`existence_error(directory,Directory)`

An owned file or directory cannot be removed:

`permission_error(delete,source_sink,Path)`

A character sequence is not valid UTF-8:

`representation_error(utf_8)`

A launcher operation fails:

`asp_transport(Stage,Status,Details)`

Solver output is invalid or inconsistent with its exit status:

`asp_solver_error(Formal,Status,Diagnostics,SourceMap)`

`program/1`

Generates and prints ASP source to the current output, equivalent to `program/2` followed by `print_program/1`, without invoking Clingo.

Compilation flags:

`static`

Template:

`program(Source)`

Mode and number of proofs:

`program(+compound)` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `asp_problem_protocol`:

`domain_error(asp_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(asp_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(asp_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:
`type_error(integer,Code)`
An element `Code` of the list `Codes` is an integer but not a valid character code:
`domain_error(character_code,Code)`
`Path` is ground but not an atom:
`type_error(atom,Path)`
`Path` does not exist:
`existence_error(source_sink,Path)`
`Path` cannot be opened for reading:
`permission_error(open,source_sink,Path)`
The current output stream does not allow writing:
`permission_error(output,stream,Stream)`

`program/2`

Returns ASP source codes from `object(Object)`, `terms(Statements)`, `codes(Codes)`, or `file(Path)` without invoking Clingo.

Compilation flags:
`static`

Template:
`program(Source,Codes)`
Mode and number of proofs:
`program(+compound,-list(integer))` - `one_or_error`

Exceptions:
`Source` is not a ground term:
`instantiation_error`
`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `asp_problem_protocol`:
`domain_error(asp_source,Source)`
`Statements` is ground but not a list:
`type_error(list,Statements)`
A statement enumerated by the provider object is not a valid statement:
`domain_error(asp_statement,Statement)`
An origin enumerated by the provider object is invalid, or its origins and statements differ in length:
`domain_error(asp_origins,Origins)`
`Codes` is ground but not a list of character codes:
`type_error(codes,Codes)`
An element `Code` of the list `Codes` is ground but not an integer:
`type_error(integer,Code)`
An element `Code` of the list `Codes` is an integer but not a valid character code:
`domain_error(character_code,Code)`
`Path` is ground but not an atom:
`type_error(atom,Path)`
`Path` does not exist:

`existence_error(source_sink,Path)`
`Path` cannot be opened for reading:
`permission_error(open,source_sink,Path)`

`print_program/1`

Prints ASP character codes to the current output without adding a newline or invoking Clingo.

Compilation flags:
`static`

Template:
`print_program(Codes)`
Mode and number of proofs:
`print_program(+list(integer))` - `one_or_error`

Exceptions:
`Codes` is not a ground term:
`instantiation_error`
`Codes` is ground but not a list of character codes:
`type_error(codes,Codes)`
An element `Code` of the list `Codes` is ground but not an integer:
`type_error(integer,Code)`
An element `Code` of the list `Codes` is an integer but not a valid character code:
`domain_error(character_code,Code)`
The current output stream does not allow writing:
`permission_error(output,stream,Stream)`

`model/2`

Enumerates, by backtracking, collected `model(Symbols, Costs, RawValues)` records without starting a process.

Compilation flags:
`static`

Template:
`model(Result,Model)`
Mode and number of proofs:
`model(+compound,-compound)` - `zero_or_more`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected Clingo result:

`domain_error(asp_result,Result)`

---

result_property/2

Enumerates, by backtracking, `status(Status)` and the stored result properties.

Compilation flags:
`static`

Template:
`result_property(Result,Property)`
Mode and number of proofs:
`result_property(+compound,?compound)` - `zero_or_more`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected Clingo result:
`domain_error(asp_result,Result)`

---

print_result/1

Prints collected answers, optimization costs, solver status, and summary metadata to the current output using Clingo-style labels, without invoking the solver.

Compilation flags:
`static`

Template:
`print_result(Result)`
Mode and number of proofs:
`print_result(+compound)` - `one_or_error`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected Clingo result:
`domain_error(asp_result,Result)`
The current output stream does not allow writing:
`permission_error(output,stream,Stream)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.10.8 asp_symbols

Parser for shown Clingo ground symbols.

Availability:
`logtalk_load(asp(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Uses:
asp_program
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses one complete ground symbol into a canonical tagged term.

Compilation flags:
`static`

Template:
`parse(Codes,Symbol)`
Mode and number of proofs:
`parse(+list(integer),-term)` - `one_or_error`

Exceptions:
`Codes` is not a ground term:
`instantiation_error`
`Codes` is ground but not a list of character codes:
`type_error(codes,Codes)`
`Codes` does not represent a supported ground symbol:
`domain_error(asp_symbol,Codes)`
A numeric symbol cannot be represented as an integer:
`representation_error(asp_integer)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.11 assertions

object

1.11.1 assertions

Proxy object for simplifying the use of the assertion meta-predicates.

Availability:
`logtalk_load(assertions(loader))`

Author: Paulo Moura
Version: 2:0:0

Date: 2014-04-03

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *assertions(_)*

Remarks:
(none)

Inherited public predicates:
assertion/1 *assertion/2* *goal_expansion/2* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.11.2 `assertions(Mode)`

A simple assertions framework. Can be used as a hook object for either suppressing assertions (`production` mode) or expanding them with file context information (`debug` mode).

Availability:
`logtalk_load(assertions(loader))`

Author: Paulo Moura
Version: 2:2:2
Date: 2022-07-04

Compilation flags:

`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
logtalk

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
  - `assertion/1`
  - `assertion/2`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`assertion/1`

Checks that an assertion is true. Uses the structured message printing mechanism for printing the results using a silent message for assertion success and a error message for assertion failure.

Compilation flags:
`static`

Template:
`assertion(Goal)`
Meta-predicate template:
`assertion(0)`
Mode and number of proofs:
`assertion(@callable)` - `one`

assertion/2

Checks that an assertion is true. Uses the structured message printing mechanism for printing the results using a silent message for assertion success and a error message for assertion failure. The context argument can be used to e.g. pass location data.

Compilation flags:
```
static
```

Template:
```
assertion(Context,Goal)
```
Meta-predicate template:
```
assertion(*,0)
```
Mode and number of proofs:
```
assertion(@term,@callable) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.11.3 assertions_messages

Assertions framework default message translations.

Availability:
```
logtalk_load(assertions(loader))
```

Author: Paulo Moura
Version: 2:2:0
Date: 2018-02-20

Compilation flags:
```
static
```

Provides:
logtalk::message_prefix_stream/4
logtalk::message_tokens//2

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.12 assignvars

object

1.12.1 `assignvars`

Assignable variables (supporting backtracable assignment of non-variable terms).

Availability:
`logtalk_load(assignvars(loader))`

Author: Nobukuni Kino and Paulo Moura
Version: 1:7:0
Date: 2018-07-11

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *assignvarsp*

Remarks:
(none)

Inherited public predicates:

(<=)/2 (=>)/2 assignable/1 assignable/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.12.2 `assignvarsp`

Assignable variables (supporting backtracable assignment of non-variable terms) protocol.

Availability:
```
logtalk_load(assignvars(loader))
```

Author: Nobukuni Kino and Paulo Moura
Version: 1:0:1
Date: 2019-06-10

Compilation flags:
```
static
```

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *assignable/1*
  - *assignable/2*
  - *(<=)/2*
  - *(=>)/2*
- *Protected predicates*
- *Private predicates*
- *Operators*
  - *op(100,xfx,<=)*
  - *op(100,xfx,=>)*

Public predicates

assignable/1

Makes `Variable` an assignable variable. Initial state will be empty.

Compilation flags:
`static`

Template:
`assignable(Variable)`
Mode and number of proofs:
`assignable(--assignvar)` - `one`

Exceptions:
Variable is not a variable:
`type_error(variable,Variable)`

---

assignable/2

Makes `Variable` an assignable variable and sets its initial state to `Value`.

Compilation flags:
`static`

Template:
`assignable(Variable,Value)`
Mode and number of proofs:
`assignable(--assignvar,@nonvar)` - `one`

Exceptions:
Variable is not a variable:
`type_error(variable,Variable)`
Value is not instantiated:
`instantiation_error`

(<=)/2

Sets the state of the assignable variable `Variable` to `Value` (initializing the variable if needed).

Compilation flags:
`static`

Template:
`Variable<=Value`

Mode and number of proofs:
`(?assignvar)<=(@nonvar)` - `one`

Exceptions:
Value is not instantiated:
`instantiation_error`

(=>)/2

Unifies `Value` with the current state of the assignable variable `Variable`.

Compilation flags:
`static`

Template:
`Variable=>Value`

Mode and number of proofs:
`+assignvar=> ?nonvar` - `zero_or_one`

Exceptions:
Variable is not instantiated:
`instantiation_error`

Protected predicates

(none)

Private predicates

(none)

Operators

`op(100,xfx,<=)`

Scope:
public

`op(100,xfx,=>)`

Scope:
public

See also

assignvars

1.13 association_rule_miner

object

1.13.1 `association_rule_miner`

Association rule miner deriving confidence- and lift-scored rules from frequent itemsets and sequential patterns.

Availability:
`logtalk_load(association_rule_miner(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Imports:
public *association_rule_miner_common*
Uses:
format
list
type

Remarks:
(none)

Inherited public predicates:

association_rule_miner_options/2 *check_association_rule_miner/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *mine/3* *mine/4* *option/2* *option/3* *print_association_rule_miner/1* *rule_metric/3* *valid_association_rule_miner/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

association_rule_miner_protocol, *pattern_miner_protocol*, *transaction_dataset_protocol*, *sequence_dataset_protocol*

category

1.13.2 `association_rule_miner_common`

Shared predicates for association rule miner diagnostics, options, validation, and export.

Availability:
`logtalk_load(association_rule_miner(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-03

Compilation flags:
`static`

Implements:
`public` *association_rule_miner_protocol*
Extends:
`public` *options*
Uses:

format
list
type

Remarks:
(none)

Inherited public predicates:
association_rule_miner_options/2 *check_association_rule_miner/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *mine/3* *mine/4* *option/2* *option/3* *print_association_rule_miner/1* *rule_metric/3* *valid_association_rule_miner/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *rule_kind/2*
  - *check_rule_options/1*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

rule_kind/2

Returns the rule kind implied by a supported source pattern miner.

Compilation flags:
`static`

Template:
`rule_kind(SourceMiner,RuleKind)`
Mode and number of proofs:
`rule_kind(+atom,-atom) - one`

---

check_rule_options/1

Checks cross-option constraints for effective association rule mining options.

Compilation flags:
`static`

Template:
`check_rule_options(Options)`

Mode and number of proofs:
`check_rule_options(+list(compound)) - one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.13.3 `association_rule_miner_protocol`

Protocol for association rule miners deriving scored rules from mined frequent patterns.

Availability:
`logtalk_load(association_rule_miner(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-03

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *mine/4*
  - *mine/3*
  - *diagnostics/2*
  - *diagnostic/2*
  - *association_rule_miner_options/2*
  - *rule_metric/3*
  - *check_association_rule_miner/1*
  - *valid_association_rule_miner/1*
  - *export_to_clauses/4*

    - *export_to_file/4*
    - *print_association_rule_miner/1*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

mine/4

Derives association rules from a mined pattern result and its dataset using the specified options.

Compilation flags:
```
static
```

Template:
```
mine(Dataset,PatternMiner,AssociationRuleMiner,Options)
```
Mode and number of proofs:
```
mine(+object_identifier,+compound,-compound,+list(compound)) - one
```

---

mine/3

Derives association rules from a mined pattern result and its dataset using default options.

Compilation flags:
```
static
```

Template:
```
mine(Dataset,PatternMiner,AssociationRuleMiner)
```
Mode and number of proofs:
```
mine(+object_identifier,+compound,-compound) - one
```

---

diagnostics/2

Returns diagnostics and metadata associated with an association rule miner result.

Compilation flags:
```
static
```

Template:
```
diagnostics(AssociationRuleMiner,Diagnostics)
```
Mode and number of proofs:
```
diagnostics(+compound,-list(compound)) - one
```

---

diagnostic/2

Tests or enumerates individual diagnostics metadata terms.

Compilation flags:
`static`

Template:
`diagnostic(AssociationRuleMiner,Diagnostic)`
Mode and number of proofs:
`diagnostic(+compound,?compound)` - `zero_or_more`

---

association_rule_miner_options/2

Returns the effective options recorded in an association rule miner result.

Compilation flags:
`static`

Template:
`association_rule_miner_options(AssociationRuleMiner,Options)`
Mode and number of proofs:
`association_rule_miner_options(+compound,-list(compound))` - `one`

---

rule_metric/3

Computes, tests, or enumerates on-demand metrics for a rule in an association rule miner result. The supported metrics are `leverage(Value)`, `jaccard(Value)`, `cosine(Value)`, and `kulczynski(Value)`.

Compilation flags:
`static`

Template:
`rule_metric(AssociationRuleMiner,Rule,Metric)`
Mode and number of proofs:
`rule_metric(+compound,+compound,?compound)` - `zero_or_more`

---

check_association_rule_miner/1

Checks that the argument is a structurally valid association rule miner result, throwing an exception otherwise.

Compilation flags:
`static`

Template:
`check_association_rule_miner(AssociationRuleMiner)`
Mode and number of proofs:
`check_association_rule_miner(@compound)` - `one_or_error`

Exceptions:
`AssociationRuleMiner` is a variable:
`instantiation_error`
`AssociationRuleMiner` is neither a variable nor a valid association rule miner result:
`domain_error(association_rule_miner,AssociationRuleMiner)`

---

valid_association_rule_miner/1

True when `check_association_rule_miner/1` succeeds for the argument without throwing an exception.

Compilation flags:
`static`

Template:
`valid_association_rule_miner(AssociationRuleMiner)`
Mode and number of proofs:
`valid_association_rule_miner(@compound)` - `zero_or_one`

---

export_to_clauses/4

Converts an association rule miner result into a list of predicate clauses.

Compilation flags:
`static`

Template:
`export_to_clauses(Dataset,AssociationRuleMiner,Functor,Clauses)`
Mode and number of proofs:
`export_to_clauses(+object_identifier,+compound,+callable,-list(clause))` - `one`

---

export_to_file/4

Exports an association rule miner result to a file.

Compilation flags:
```
static
```

Template:
```
export_to_file(Dataset,AssociationRuleMiner,Functor,File)
```
Mode and number of proofs:
```
export_to_file(+object_identifier,+compound,+callable,+atom) - one
```

---

print_association_rule_miner/1

Prints an association rule miner result to the current output stream in a human-readable format.

Compilation flags:
```
static
```

Template:
```
print_association_rule_miner(AssociationRuleMiner)
```
Mode and number of proofs:
```
print_association_rule_miner(+compound) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

pattern_miner_protocol, *transaction_dataset_protocol*, *sequence_dataset_protocol*

1.14 atms

object

1.14.1 `atms`

Portable incremental Assumption-based Truth Maintenance System.

Availability:
`logtalk_load(atms(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-18

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *atms_protocol*
Uses:
avltree
list

Remarks:

- Algorithm: A label is an antichain of minimal environments. New label environments are propagated as deltas through a reverse justification index.
- Nogoods: Nogoods are also maintained as an antichain and immediately prune every affected label.
- Retraction: General removals rebuild derived state from surviving assumptions and stored justifications. Batch removals rebuild at most once; isolated non-assumption node removals preserve the other derived indexes.
- Portability: The implementation is purely functional and uses the portable AVL tree dictionary for its state indexes.

Inherited public predicates:
assumption/2 assumptions/2 clear/2 consistent/2 contradiction/2 create_assumption/4 create_contradiction/4 create_node/4 in_label/3 interpretations/2 justifications/2 justify/5 label/3 new/1 new/2 node/2 nodes/2 nogood/2 representation/2 retract_batch/4 retract_justification/5 retract_justifications/3 retract_node/3 retract_nodes/3 why/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

atms_environment_protocol, *atms_ordered_list_environment*, *atms_bitset_environment*

object

1.14.2 `atms_bitset_environment`

ATMS environments represented by non-negative integer bitsets. Best suited to a modest number of assumptions.

Availability:
`logtalk_load(atms(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-18

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *atms_environment_protocol*

Remarks:
(none)

Inherited public predicates:
empty/1 *equal/2* *from_list/2* *singleton/2* *subset/2* *to_list/2* *union/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.14.3 atms_environment_protocol

Protocol for the opaque environment representation used by the ATMS library.

Availability:
`logtalk_load(atms(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-26

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *empty/1*
  - *singleton/2*
  - *union/3*
  - *subset/2*
  - *equal/2*
  - *from_list/2*
  - *to_list/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

empty/1

Returns the empty environment.

Compilation flags:
```
static
```

Template:
```
empty(Environment)
```
Mode and number of proofs:
```
empty(-atms_environment) - one
```

---

singleton/2

Returns the environment containing only the given node.

Compilation flags:
```
static
```

Template:
```
singleton(Node,Environment)
```
Mode and number of proofs:
```
singleton(+atms_node,-atms_environment) - one
```

---

union/3

Returns the union of two environments.

Compilation flags:
```
static
```

Template:
```
union(Environment1,Environment2,Union)
```
Mode and number of proofs:
```
union(+atms_environment,+atms_environment,-atms_environment) - one
```

---

subset/2

True iff the first environment is a subset of the second environment.

Compilation flags:
```
static
```

Template:
```
subset(Environment1,Environment2)
```
Mode and number of proofs:
```
subset(+atms_environment,+atms_environment) - zero_or_one
```

---

equal/2

True iff the two environments are equal.

Compilation flags:
```
static
```

Template:
```
equal(Environment1,Environment2)
```
Mode and number of proofs:
```
equal(+atms_environment,+atms_environment) - zero_or_one
```

---

from_list/2

Converts a list of node identifiers to the canonical internal environment representation.

Compilation flags:
```
static
```

Template:
```
from_list(Nodes,Environment)
```
Mode and number of proofs:
```
from_list(+list(atms_node),-atms_environment) - one
```

---

to_list/2

Converts an internal environment to an ordered list of node identifiers.

Compilation flags:
`static`

Template:
`to_list(Environment,Nodes)`
Mode and number of proofs:
`to_list(+atms_environment,-list(atms_node))` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

atms, *atms_ordered_list_environment*, *atms_bitset_environment*, *atms_segmented_bitset_environment*

object

1.14.4 atms_ordered_list_environment

ATMS environments represented as canonical ordered lists of node identifiers.

Availability:
`logtalk_load(atms(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-26

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *atms_environment_protocol*

Remarks:
(none)

Inherited public predicates:

empty/1 *equal/2* *from_list/2* *singleton/2* *subset/2* *to_list/2* *union/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.14.5 `atms_protocol`

Portable, functional Assumption-based Truth Maintenance System API.

Availability:

```
logtalk_load(atms(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-01

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

- State: All update predicates take an input ATMS state and return a new state. This permits several ATMS instances without relying on backend-specific mutable databases.
- Retraction: Node and batch justification removal is idempotent. Single justification removal deletes one matching occurrence. Earlier states and surviving identifiers are preserved.
- Environments: The public API uses ordered lists of node identifiers. The representation selected when creating a state is internal.

Inherited public predicates:
(none)

- *Public predicates*
  - *new/1*
  - *new/2*
  - *clear/2*
  - *representation/2*
  - *create_node/4*
  - *create_assumption/4*
  - *create_contradiction/4*
  - *justify/5*
  - *retract_justification/5*
  - *retract_justifications/3*
  - *retract_node/3*
  - *retract_nodes/3*
  - *retract_batch/4*
  - *node/2*
  - *assumption/2*
  - *contradiction/2*
  - *nodes/2*
  - *assumptions/2*
  - *justifications/2*
  - *label/3*
  - *in_label/3*
  - *consistent/2*
  - *nogood/2*
  - *interpretations/2*
  - *why/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new/1

Creates a new empty ATMS using the default ordered-list environment representation.

Compilation flags:
`static`

Template:
`new(State)`

Mode and number of proofs:
`new(-atms_state) - one`

new/2

Creates a new empty ATMS using the specified environment representation object. Fails if the object does not conform to the environment protocol.

Compilation flags:
```
static
```

Template:
```
new(Representation,State)
```
Mode and number of proofs:
```
new(+object,-atms_state) - zero_or_one
```

---

clear/2

Returns a new empty ATMS using the same environment representation as the given state.

Compilation flags:
```
static
```

Template:
```
clear(State,ClearedState)
```
Mode and number of proofs:
```
clear(+atms_state,-atms_state) - one
```

---

representation/2

Returns the environment representation object used by the given state.

Compilation flags:
```
static
```

Template:
```
representation(State,Representation)
```
Mode and number of proofs:
```
representation(+atms_state,?object) - one
```

---

create_node/4

Creates an ordinary node for the given datum and returns the updated state.

Compilation flags:
```
static
```

Template:
```
create_node(Datum,State,Node,NewState)
```
Mode and number of proofs:
```
create_node(+term,+atms_state,-atms_node,-atms_state) - one
```

---

create_assumption/4

Creates an assumption node for the given datum and returns the updated state.

Compilation flags:
```
static
```

Template:
```
create_assumption(Datum,State,Assumption,NewState)
```
Mode and number of proofs:
```
create_assumption(+term,+atms_state,-atms_node,-atms_state) - one
```

---

create_contradiction/4

Creates a contradiction node for the given datum and returns the updated state.

Compilation flags:
```
static
```

Template:
```
create_contradiction(Datum,State,Contradiction,NewState)
```
Mode and number of proofs:
```
create_contradiction(+term,+atms_state,-atms_node,-atms_state) - one
```

---

justify/5

Adds a Horn justification from the antecedent nodes to the consequent node, incrementally propagating label changes. Fails if any referenced node is unknown.

Compilation flags:

```
static
```

Template:

```
justify(Consequent,Antecedents,Info,State,NewState)
```

Mode and number of proofs:

```
justify(+atms_node,+list(atms_node),+term,+atms_state,-atms_state) - zero_or_one
```

---

retract_justification/5

Removes one occurrence of the specified Horn justification, rebuilding labels and nogoods when needed. Returns the input state unchanged if the justification is absent.

Compilation flags:

```
static
```

Template:

```
retract_justification(Consequent,Antecedents,Info,State,NewState)
```

Mode and number of proofs:

```
retract_justification(+atms_node,+list(atms_node),+term,+atms_state,-atms_state) -
one
```

Remarks:

- Identity: Antecedents are sorted and deduplicated as in `justify/5`. The complete justification, including Info, is matched using term identity rather than unification.
- Occurrences: If variable aliasing has made distinct stored rules identical, removes the most recently added matching occurrence. Use `retract_justifications/3` to remove all matches.
- Redundant rules: If another rule with the same consequent and canonical antecedents remains, preserves labels and nogoods and removes only the stored rule and its reverse dependency entries. Info is ignored only for this redundancy check.

---

retract_justifications/3

Removes all occurrences matching the requested justification terms, rebuilding derived state at most once. Returns the input state unchanged if none match.

Compilation flags:

```
static
```

Template:

```
retract_justifications(Justifications,State,NewState)
```

Mode and number of proofs:

```
retract_justifications(+list(term),+atms_state,-atms_state) - one
```

Remarks:

- Targets: Each target is a `justification(Consequent, Antecedents, Info)` term. Antecedents are sorted and deduplicated; the full term is matched using term identity.
- Idempotence: Missing targets and duplicate requests have no additional effect. All matching stored occurrences are removed, including those made identical by variable aliasing.
- Redundant rules: Preserves labels and nogoods if every removed consequent and canonical antecedent signature still has a surviving rule, regardless of Info. Reverse dependencies retain the surviving occurrence counts.

---

retract_node/3

Removes an ordinary, assumption, or contradiction node and all justifications mentioning it, rebuilding derived state when needed. Returns the input state unchanged if the node is absent.

Compilation flags:

```
static
```

Template:

```
retract_node(Node,State,NewState)
```

Mode and number of proofs:

```
retract_node(+atms_node,+atms_state,-atms_state) - one
```

Remarks:

- Dependents: Other nodes are retained even when they lose all support.
- Isolated nodes: Removing a non-assumption node that appears in no justification deletes its label entry without rebuilding other labels, nogoods, or dependency indexes.
- Identifiers: Surviving node identifiers are preserved and removed identifiers are not reused in descendant states.

---

retract_nodes/3

Removes the requested nodes and every justification mentioning any of them, rebuilding derived state at most once. Returns the input state unchanged if no nodes are removed.

Compilation flags:

```
static
```

Template:

```
retract_nodes(Nodes,State,NewState)
```

Mode and number of proofs:

```
retract_nodes(+list(atms_node),+atms_state,-atms_state) - one
```

Remarks:

- Targets: Missing nodes and duplicate requests are ignored. All other nodes are retained, even when they lose all support.

- Isolated nodes: If no assumption or incident justification is removed, only node, role, and label entries are deleted; other derived indexes are preserved.
- Identifiers: Surviving identifiers and the next node identifier are preserved.

---

retract_batch/4

Removes requested nodes, their incident justifications, and all exact matches for requested justification terms in one batch, rebuilding derived state at most once.

Compilation flags:

```
static
```

Template:

```
retract_batch(Nodes,Justifications,State,NewState)
```

Mode and number of proofs:

```
retract_batch(+list(atms_node),+list(term),+atms_state,-atms_state) - one
```

Remarks:

- Targets: Uses the same canonical antecedents and complete term identity as `retract_justifications/3`. Missing targets and duplicate or overlapping requests are ignored.
- Occurrences: Every matching stored occurrence is removed, including occurrences made identical by variable aliasing.
- Identifiers: Surviving nodes, identifiers, and rule insertion order are preserved. Other nodes remain even when they lose support.
- No changes: Returns the input state unchanged if no targets match. Isolated non-assumption node removal preserves the other derived indexes.
- Redundant rules: If no nodes are removed, uses the same redundant-rule fast path as `retract_justifications/3`. Effective node removal affecting assumptions or rules rebuilds once.

---

node/2

Enumerates or tests the nodes in the given state.

Compilation flags:

```
static
```

Template:

```
node(Node,State)
```

Mode and number of proofs:

```
node(?atms_node,+atms_state) - zero_or_more
```

---

assumption/2

Enumerates or tests the assumption nodes in the given state.

Compilation flags:
`static`

Template:
`assumption(Assumption,State)`
Mode and number of proofs:
`assumption(?atms_node,+atms_state)` - `zero_or_more`

---

contradiction/2

Enumerates or tests the contradiction nodes in the given state.

Compilation flags:
`static`

Template:
`contradiction(Contradiction,State)`
Mode and number of proofs:
`contradiction(?atms_node,+atms_state)` - `zero_or_more`

---

nodes/2

Returns the node-datum pairs in node identifier order.

Compilation flags:
`static`

Template:
`nodes(State,Nodes)`
Mode and number of proofs:
`nodes(+atms_state,-list(pair))` - `one`

---

assumptions/2

Returns the assumption nodes in node identifier order.

Compilation flags:
```
static
```

Template:
```
assumptions(State,Assumptions)
```
Mode and number of proofs:
```
assumptions(+atms_state,-list(atms_node)) - one
```

---

justifications/2

Returns all justifications in insertion order.

Compilation flags:
```
static
```

Template:
```
justifications(State,Justifications)
```
Mode and number of proofs:
```
justifications(+atms_state,-list(term)) - one
```

---

label/3

Returns the minimal consistent environments supporting the given node. Fails if the node is unknown.

Compilation flags:
```
static
```

Template:
```
label(Node,State,Environments)
```
Mode and number of proofs:
```
label(+atms_node,+atms_state,-list(list(atms_node))) - zero_or_one
```

---

in_label/3

True iff the given environment is a member of the node label.

Compilation flags:
`static`

Template:
`in_label(Node,Environment,State)`
Mode and number of proofs:
`in_label(+atms_node,+list(atms_node),+atms_state)` - `zero_or_one`

---

consistent/2

True iff the given environment contains no recorded nogood.

Compilation flags:
`static`

Template:
`consistent(Environment,State)`
Mode and number of proofs:
`consistent(+list(atms_node),+atms_state)` - `zero_or_one`

---

nogood/2

Enumerates or tests the minimal nogood environments in the given state.

Compilation flags:
`static`

Template:
`nogood(Environment,State)`
Mode and number of proofs:
`nogood(?list(atms_node),+atms_state)` - `zero_or_more`

---

interpretations/2

Returns the maximal consistent environments over all assumptions.

Compilation flags:
```
static
```

Template:
```
interpretations(State,Interpretations)
```
Mode and number of proofs:
```
interpretations(+atms_state,-list(list(atms_node))) - one
```

---

why/3

Returns the direct justifications for the given node in insertion order.

Compilation flags:
```
static
```

Template:
```
why(Node,State,Justifications)
```
Mode and number of proofs:
```
why(+atms_node,+atms_state,-list(term)) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.14.6 atms_segmented_bitset_environment

ATMS environments represented as sparse ordered lists of nonempty 16-bit blocks.

Availability:
```
logtalk_load(atms(loader))
```

Author: Paulo Moura
Version: 1:0:0

Date: 2026-09-26

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *atms_environment_protocol*

Remarks:
(none)

Inherited public predicates:
empty/1 equal/2 from_list/2 singleton/2 subset/2 to_list/2 union/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.15 avro

object

1.15.1 `avro`

Apache Avro binary format parser and generator.

Availability:
`logtalk_load(avro(loader))`

Author: Paulo Moura
Version: 1:2:1

Date: 2026-06-14

Compilation flags:
`static, context_switching_calls`

Uses:
ieee_754(Precision,ByteOrder,NaNRepresentation)
json(ObjectRepresentation,PairRepresentation,StringRepresentation)
list
reader

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *parse/3*
  - *generate/3*
  - *generate/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses Avro binary data from the given source (`bytes(List)`, `stream(Stream)`, or `file(Path)`) returning a `Schema-Data` pair. When the schema is not present in the file, `Schema` is unified with `false`.

Compilation flags:
`static`

Template:
`parse(Source,Schema-Data)`

Mode and number of proofs:
`parse(++compound,--pair)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid Avro source term:
`domain_error(avro_source,Source)`

parse/3

Parses Avro binary data from the given source using the provided schema, returning the decoded data.

Compilation flags:
`static`

Template:
`parse(Source,Schema,Data)`
Mode and number of proofs:
`parse(++compound,++term,--term)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid Avro source term:
`domain_error(avro_source,Source)`

---

generate/3

Generates Avro binary data to the given sink (`bytes(List)`, `stream(Stream)`, or `file(Path)`) from the given schema and data. The schema is not included in the output.

Compilation flags:
`static`

Template:
`generate(Sink,Schema,Data)`
Mode and number of proofs:
`generate(++compound,++term,++term)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid Avro sink term:
`domain_error(avro_sink,Sink)`

---

generate/4

Generates Avro binary data to the given sink from the given schema and data. When `IncludeSchema` is `true`, generates an Avro Object Container File with the schema embedded.

Compilation flags:
`static`

Template:

```
generate(Sink,IncludeSchema,Schema,Data)
```

Mode and number of proofs:

```
generate(++compound,++boolean,++term,++term) - one_or_error
```

Exceptions:

`Sink` is a variable:

```
instantiation_error
```

`Sink` is neither a variable nor a valid Avro sink term:

```
domain_error(avro_sink,Sink)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.16 base32

object

1.16.1 `base32`

Base32 encoder and decoder (RFC 4648).

Availability:

```
logtalk_load(base32(loader))
```

Author: Paulo Moura
Version: 1:0:2
Date: 2026-07-20

Compilation flags:

```
static, context_switching_calls
```

Uses:

reader

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses Base32 data from the given source (`atom(Atom)`, `chars(List)`, `codes(List)`, `stream(Stream)`, or `file(Path)`) into a list of bytes.

Compilation flags:
`static`

Template:
`parse(Source,Bytes)`

Mode and number of proofs:
`parse(++compound,--list(byte))` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid Base32 source term:
`domain_error(base32_source,Source)`

---

generate/2

Generates Base32 in the representation specified in the first argument (`atom(Atom)`, `chars(List)`, `codes(List)`, `stream(Stream)`, or `file(Path)`) for the list of bytes in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,Bytes)`

Mode and number of proofs:
`generate(+compound,+list(byte))` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid Base32 sink term:

```
domain_error(base32_sink,Sink)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.17 base58

object

1.17.1 `base58`

Base58 encoder and decoder (Bitcoin alphabet variant).

Availability:
`logtalk_load(base58(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-06-14

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses Base58 data from the given source (`atom(Atom)`, `chars(List)`, or `codes(List)`) into a list of bytes.

Compilation flags:
`static`

Template:
`parse(Source,Bytes)`
Mode and number of proofs:
`parse(++compound,--list(byte))` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid Base58 source term:
`domain_error(base58_source,Source)`
`Source` contains Base58 data with characters outside the Base58 alphabet:
`representation_error(base58)`

---

generate/2

Generates Base58 in the representation specified in the first argument (`atom(Atom)`, `chars(List)`, or `codes(List)`) for the list of bytes in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,Bytes)`
Mode and number of proofs:
`generate(+compound,+list(byte))` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid Base58 sink term:
`domain_error(base58_sink,Sink)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.18 base64

object

1.18.1 `base64`

Base64 parser and generator.

Availability:
`logtalk_load(base64(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-06-14

Compilation flags:
`static, context_switching_calls`

Uses:
reader

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses the Base64 data from the given source (`atom(Atom)`, `chars(List)`, `codes(List)`, `stream(Stream)`, or `file(Path)` into a list of bytes.

Compilation flags:
`static`

Template:
`parse(Source,Bytes)`
Mode and number of proofs:
`parse(++compound,--list(byte))` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid Base64 source term:
`domain_error(base64_source,Source)`
`Source` contains Base64 data with characters outside the Base64 alphabet:
`representation_error(base64)`

---

generate/2

Generates Base64 in the representation specified in the first argument (`atom(Atom)`, `chars(List)`, `codes(List)`, `stream(Stream)`, or `file(Path)` for the list of bytes in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,Bytes)`
Mode and number of proofs:
`generate(+compound,+list(byte))` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid Base64 sink term:
`domain_error(base64_sink,Sink)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.18.2 `base64url`

Base64URL parser and generator.

Availability:
`logtalk_load(base64(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses the Base64URL data from the given source (`atom(Atom)`, `chars(List)`, or `codes(List)` into a URL (using the same format as the source).

Compilation flags:

`static`

Template:

`parse(Source,URL)`

Mode and number of proofs:

`parse(++compound,--types([atom,chars,codes]))` - `one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid Base64 source term:

`domain_error(base64url_source,Source)`

`Source` contains Base64URL data with characters outside the Base64URL alphabet:

`representation_error(base64)`

---

generate/2

Generates Base64URL data in the representation specified in the first argument (`atom(Atom)`, `chars(List)`, or `codes(List)` for the given URL (given in the same format as the sink).

Compilation flags:

`static`

Template:

`generate(Sink,URL)`

Mode and number of proofs:

`generate(+compound,+types([atom,chars,codes]))` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid Base64 sink term:

`domain_error(base64url_sink,Sink)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.18.3 `base64url_no_padding`

Base64URL parser and generator without padding.

Availability:
`logtalk_load(base64(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses the unpadded Base64URL data from the given source (`atom(Atom)`, `chars(List)`, or `codes(List)` into a URL (using the same format as the source).

Compilation flags:

`static`

Template:

`parse(Source,URL)`

Mode and number of proofs:

`parse(++compound,--types([atom,chars,codes]))` - `one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid Base64URL source term:

`domain_error(base64url_source,Source)`

`Source` contains Base64URL data with characters outside the Base64URL alphabet:

`representation_error(base64)`

---

generate/2

Generates unpadded Base64URL data in the representation specified in the first argument (`atom(Atom)`, `chars(List)`, or `codes(List)` for the given URL (given in the same format as the sink).

Compilation flags:

`static`

Template:

`generate(Sink,URL)`

Mode and number of proofs:

`generate(+compound,+types([atom,chars,codes]))` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid Base64URL sink term:

`domain_error(base64url_sink,Sink)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.19 base85

object

1.19.1 `base85`

Base85 encoder and decoder (Ascii85/btoa variant).

Availability:
`logtalk_load(base85(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-06-14

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses Base85 (Ascii85) data from the given source (`atom(Atom)`, `chars(List)`, or `codes(List)`) into a list of bytes.

Compilation flags:
`static`

Template:
`parse(Source,Bytes)`
Mode and number of proofs:
`parse(++compound,--list(byte))` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid Base85 source term:
`domain_error(base85_source,Source)`
`Source` contains Base85 data with characters outside the Base85 alphabet:
`representation_error(base85)`

---

generate/2

Generates Base85 (Ascii85) in the representation specified in the first argument (`atom(Atom)`, `chars(List)`, or `codes(List)`) for the list of bytes in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,Bytes)`
Mode and number of proofs:
`generate(+compound,+list(byte))` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid Base85 sink term:
`domain_error(base85_sink,Sink)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.20 baseline_forecasting

object

1.20.1 `baseline_forecasting`

Naive, seasonal naive, mean, and drift time series forecasters with missing observations and immutable online updates.

Availability:
`logtalk_load(baseline_forecasting(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-14

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *forecaster_common*

Uses:
format
list
type

Remarks:
(none)

Inherited public predicates:
check_forecaster/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *forecast/3* *forecaster_options/2* *learn/2* *learn/3* *option/2* *option/3* *print_forecaster/1* *valid_forecaster/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *update/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

update/3

Returns a new forecaster after appending a numeric or missing (unbound) observation. The original forecaster is not modified.

Compilation flags:
```
static
```

Template:
```
update(Forecaster,Observation,UpdatedForecaster)
```
Mode and number of proofs:
```
update(+compound,@term,-compound) - one_or_error
```

Exceptions:

`Forecaster` is a variable:
```
instantiation_error
```
`Forecaster` is not a valid baseline forecaster:
```
domain_error(forecaster,Forecaster)
```
`Observation` is neither a variable nor a number:
```
type_error(number,Observation)
```
Updating the numeric sum raises an arithmetic evaluation error:
```
evaluation_error(Error)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.21 bayesian_ridge_regression

object

1.21.1 bayesian_ridge_regression

Bayesian ridge regression regressor supporting continuous and mixed-feature datasets using evidence maximization and posterior uncertainty over coefficients. Learns from a dataset object implementing the `regression_dataset_protocol` protocol and returns a regressor term that can be used for prediction, predictive-distribution queries, and export as predicate clauses.

Availability:
```
logtalk_load(bayesian_ridge_regression(loader))
```

Author: Paulo Moura
Version: 1:0:2
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *regressor_common*

Uses:
format
linear_algebra
list
numberlist
population
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
  - *predict_distribution/3*
  - *weight_variances/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

predict_distribution/3

Predicts the posterior predictive Gaussian distribution for a new instance using the learned regressor. The returned term has the shape `gaussian(Mean, Variance)` where `Variance` includes the learned observation noise variance and coefficient posterior uncertainty only; the intercept is not treated as a probabilistic parameter.

Compilation flags:
`static`

Template:
`predict_distribution(Regressor,Instance,Distribution)`
Mode and number of proofs:
`predict_distribution(+compound,+list,-compound)` - `one`

`weight_variances/2`

Returns the posterior marginal variances of the encoded feature coefficients in encoder order. Encoded features dropped from fitting because they had zero variance are reported with posterior variance zero.

Compilation flags:
```
static
```

Template:
```
weight_variances(Regressor,Variances)
```
Mode and number of proofs:
```
weight_variances(+compound,-list(float)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

linear_regression, *ridge_regression*, *lasso_regression*, *elastic_net_regression*, *gaussian_process_regression*, *knn_regression*, *regression_tree*, *random_forest_regression*, *gradient_boosting_regression*

1.22 block_ciphers

object

1.22.1 `aes128`

AES-128 block cipher.

Availability:
```
logtalk_load(block_ciphers(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-03

Compilation flags:
```
static, context_switching_calls
```

Extends:

`public` *aes_common(16,4,10)*

Remarks:
(none)

Inherited public predicates:
block_size/1 *decrypt_block/3* *decrypt_prepared_block/3* *encrypt_block/3* *encrypt_prepared_block/3* *key_size/1* *prepare_key/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

aes192, *aes256*, *ecb*, *cbc*, *ctr*

object

1.22.2 `aes192`

AES-192 block cipher.

Availability:
`logtalk_load(block_ciphers(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-03

Compilation flags:
`static, context_switching_calls`

Extends:

`public` *aes_common(24,6,12)*

Remarks:

(none)

Inherited public predicates:

block_size/1 *decrypt_block/3* *decrypt_prepared_block/3* *encrypt_block/3* *encrypt_prepared_block/3* *key_size/1* *prepare_key/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

aes128, *aes256*, *ecb*, *cbc*, *ctr*

object

1.22.3 `aes256`

AES-256 block cipher.

Availability:

`logtalk_load(block_ciphers(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-03

Compilation flags:

`static, context_switching_calls`

Extends:

`public` *aes_common(32,8,14)*

Remarks:

(none)

Inherited public predicates:

block_size/1 *decrypt_block/3* *decrypt_prepared_block/3* *encrypt_block/3* *encrypt_prepared_block/3* *key_size/1* *prepare_key/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

aes128, *aes192*, *ecb*, *cbc*, *ctr*

object

1.22.4 `aes_common(KeySize,Nk,Nr)`

- `KeySize` - Key size in bytes.
- `Nk` - Key size in 32-bit words.
- `Nr` - Number of encryption rounds.

AES block cipher implementation shared by the AES-128, AES-192, and AES-256 objects.

Availability:

`logtalk_load(block_ciphers(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-08-03

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *block_cipher_prepared_key_protocol*
Uses:
list
type

Remarks:
(none)

Inherited public predicates:
block_size/1 decrypt_block/3 decrypt_prepared_block/3 encrypt_block/3 encrypt_prepared_block/3 key_size/1 prepare_key/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.22.5 block_cipher_modes_common

Shared validation, block, XOR, and PKCS#7 predicates for block cipher modes.

Availability:
`logtalk_load(block_ciphers(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static`

Uses:
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *prepare_cipher/4*
  - *check_bytes/1*
  - *check_block/2*
  - *check_aligned/2*
  - *blocks/3*
  - *xor_bytes/3*
  - *pkcs7_pad/3*
  - *pkcs7_unpad/4*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

prepare_cipher/4

Validates a cipher object, queries its block size, and prepares a key.

Compilation flags:
`static`

Template:
`prepare_cipher(Cipher,Key,BlockSize,PreparedKey)`
Mode and number of proofs:
`prepare_cipher(+object_identifier,+list(byte),--positive_integer,--compound)` - `one_or_error`

Exceptions:
`Cipher` is a variable:
`instantiation_error`
`Cipher` is not a variable but does not implement the prepared-key block cipher protocol:
`domain_error(block_cipher,Cipher)`
`Cipher` implements he prepared-key block cipher protocol but reports an invalid block size:
`domain_error(block_cipher_block_size,BlockSize)`
`Key` is invalid for `Cipher`:
`type_error(list(byte,KeySize),Key)`

---

check_bytes/1

Checks a list of bytes.

Compilation flags:
`static`

Template:
`check_bytes(Bytes)`
Mode and number of proofs:
`check_bytes(+list(byte))` - `one_or_error`

Exceptions:
`Bytes` is a variable or a partial list:
`instantiation_error`
`Bytes` is neither a partial list nor a list:
`type_error(list,Bytes)`
An element `Byte` of the `Bytes` list is neither a variable nor an integer:
`type_error(integer,Byte)`

An element `Byte` of the `Bytes` list is an integer but not a valid byte:
`domain_error(byte,Byte)`

---

check_block/2

Checks a byte list whose length is exactly one block.

Compilation flags:
`static`

Template:
`check_block(BlockSize,Block)`
Mode and number of proofs:
`check_block(+positive_integer,+list(byte))` - `one_or_error`

Exceptions:
`BlockSize` is a variable:
`instantiation_error`
`BlockSize` is neither a variable nor an integer:
`type_error(integer,BlockSize)`
`BlockSize` is an integer but not a positive integer:
`domain_error(positive_integer,BlockSize)`
`Block` is a variable or a partial list:
`instantiation_error`
`Block` is not a list of exactly `BlockSize` bytes:
`type_error(list(byte,BlockSize),Block)`
An element `Byte` of the *Block*` list is neither a variable nor an integer:
`type_error(integer,Byte)`
An element `Byte` of the `Block` list is an integer but not a valid byte:
`domain_error(byte,Byte)`

---

check_aligned/2

Checks that a byte-list length is a multiple of the block size.

Compilation flags:
`static`

Template:
`check_aligned(BlockSize,Bytes)`
Mode and number of proofs:
`check_aligned(+positive_integer,+list(byte))` - `one_or_error`

Exceptions:

`BlockSize` is a variable:
```
instantiation_error
```
`BlockSize` is neither a variable nor an integer:
```
type_error(integer,BlockSize)
```
`BlockSize` is an integer but not a positive integer:
```
domain_error(positive_integer,BlockSize)
```
`Bytes` length is not a multiple of `BlockSize`:
```
domain_error(block_aligned_byte_length(BlockSize),Bytes)
```

---

blocks/3

Splits an aligned byte list into blocks.

Compilation flags:
```
static
```

Template:
```
blocks(BlockSize,Bytes,Blocks)
```
Mode and number of proofs:
```
blocks(+positive_integer,+list(byte),--list(list(byte))) - one
```

---

xor_bytes/3

XORs equal-length byte lists.

Compilation flags:
```
static
```

Template:
```
xor_bytes(Bytes1,Bytes2,Bytes)
```
Mode and number of proofs:
```
xor_bytes(+list(byte),+list(byte),--list(byte)) - one
```

---

pkcs7_pad/3

Pads bytes using PKCS#7.

Compilation flags:
```
static
```

Template:
```
pkcs7_pad(BlockSize,Bytes,PaddedBytes)
```

Mode and number of proofs:
`pkcs7_pad(+positive_integer,+list(byte),--list(byte))` - `one_or_error`

Exceptions:
`BlockSize` exceeds the maximum PKCS#7 block size:
`domain_error(pkcs7_block_size,BlockSize)`

`pkcs7_unpad/4`

Validates and removes PKCS#7 padding, reporting the supplied error value when padding is invalid.

Compilation flags:
`static`

Template:
`pkcs7_unpad(BlockSize,PaddedBytes,ErrorValue,Bytes)`
Mode and number of proofs:
`pkcs7_unpad(+positive_integer,+list(byte),+term,--list(byte))` - `one_or_error`

Exceptions:
`BlockSize` exceeds the maximum PKCS#7 block size:
`domain_error(pkcs7_block_size,BlockSize)`
`PaddedBytes` does not contain valid PKCS#7 padding:
`domain_error(pkcs7_padding,ErrorValue)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.22.6 `block_cipher_prepared_key_protocol`

Protocol for block ciphers supporting opaque prepared keys for repeated block operations.

Availability:
`logtalk_load(block_ciphers(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:

`static`

Extends:
`public` *block_cipher_protocol*

Remarks:
(none)

Inherited public predicates:
block_size/1 *decrypt_block/3* *encrypt_block/3* *key_size/1*

- *Public predicates*
  - *prepare_key/2*
  - *encrypt_prepared_block/3*
  - *decrypt_prepared_block/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

prepare_key/2

Validates and prepares a key for repeated block encryption or decryption.

Compilation flags:
`static`

Template:
`prepare_key(Key,PreparedKey)`
Mode and number of proofs:
`prepare_key(+list(byte),--compound)` - `one_or_error`

Exceptions:
`Key` is a variable or a partial list:
`instantiation_error`
`Key` is a list but not with the required number of bytes:
`type_error(list(byte,KeySize),Key)`
`Key` contains a non-integer byte:
`type_error(integer,Byte)`
`Key` contains an integer outside the byte range:
`domain_error(byte,Byte)`

encrypt_prepared_block/3

Encrypts a plaintext block using an opaque prepared key created by the same cipher object.

Compilation flags:
`static`

Template:
`encrypt_prepared_block(PreparedKey,PlaintextBlock,CiphertextBlock)`
Mode and number of proofs:
`encrypt_prepared_block(+compound,+list(byte),--list(byte))` - `one_or_error`

Exceptions:
`PreparedKey` was not created by this cipher object:
`domain_error(block_cipher_prepared_key,PreparedKey)`
`PlaintextBlock` is a variable or a partial list:
`instantiation_error`
`PlaintextBlock` is not a list of the required number of bytes:
`type_error(list(byte,BlockSize),PlaintextBlock)`
`PlaintextBlock` contains a non-integer byte:
`type_error(integer,Byte)`
`PlaintextBlock` contains an integer outside the byte range:
`domain_error(byte,Byte)`

---

decrypt_prepared_block/3

Decrypts a ciphertext block using an opaque prepared key created by the same cipher object.

Compilation flags:
`static`

Template:
`decrypt_prepared_block(PreparedKey,CiphertextBlock,PlaintextBlock)`
Mode and number of proofs:
`decrypt_prepared_block(+compound,+list(byte),--list(byte))` - `one_or_error`

Exceptions:
`PreparedKey` was not created by this cipher object:
`domain_error(block_cipher_prepared_key,PreparedKey)`
`CiphertextBlock` is a variable or a partial list:
`instantiation_error`
`CiphertextBlock` is not a list of the required number of bytes:
`type_error(list(byte,BlockSize),CiphertextBlock)`
`CiphertextBlock` contains a non-integer byte:
`type_error(integer,Byte)`
`CiphertextBlock` contains an integer outside the byte range:
`domain_error(byte,Byte)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.22.7 `block_cipher_protocol`

Protocol for one-shot encryption and decryption using a block cipher.

Availability:
`logtalk_load(block_ciphers(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-03

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *encrypt_block/3*
  - *decrypt_block/3*
  - *block_size/1*
  - *key_size/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

encrypt_block/3

Encrypts a plaintext block using the given key.

Compilation flags:
`static`

Template:
`encrypt_block(Key,PlaintextBlock,CiphertextBlock)`
Mode and number of proofs:
`encrypt_block(+list(byte),+list(byte),--list(byte))` - `one_or_error`

Exceptions:
`Key` or `PlaintextBlock` is a variable or a partial list:
`instantiation_error`
`Key` is neither a variable nor a list of the required number of bytes:
`type_error(list(byte,KeySize),Key)`
`PlaintextBlock` is neither a variable nor a list of the required number of bytes:
`type_error(list(byte,BlockSize),PlaintextBlock)`
`Key` or `PlaintextBlock` contains a non-integer byte:
`type_error(integer,Byte)`
`Key` or `PlaintextBlock` contains an integer outside the byte range:
`domain_error(byte,Byte)`

---

decrypt_block/3

Decrypts a ciphertext block using the given key.

Compilation flags:
`static`

Template:
`decrypt_block(Key,CiphertextBlock,PlaintextBlock)`
Mode and number of proofs:
`decrypt_block(+list(byte),+list(byte),--list(byte))` - `one_or_error`

Exceptions:
`Key` or `CiphertextBlock` is a variable or a partial list:
`instantiation_error`
`Key` is neither a variable nor a list of the required number of bytes:
`type_error(list(byte,KeySize),Key)`
`CiphertextBlock` is neither a variable nor a list of the required number of bytes:
`type_error(list(byte,BlockSize),CiphertextBlock)`
`Key` or `CiphertextBlock` contains a non-integer byte:
`type_error(integer,Byte)`
`Key` or `CiphertextBlock` contains an integer outside the byte range:

```
domain_error(byte,Byte)
```

---

block_size/1

Returns the cipher block size in bytes.

Compilation flags:
```
static
```

Template:
```
block_size(BlockSize)
```
Mode and number of proofs:
```
block_size(--integer) - one
```

---

key_size/1

Returns the cipher key size in bytes.

Compilation flags:
```
static
```

Template:
```
key_size(KeySize)
```
Mode and number of proofs:
```
key_size(--integer) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.22.8 cbc

Generic Cipher Block Chaining mode encryption and decryption.

Availability:
`logtalk_load(block_ciphers(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *block_cipher_modes_common*
Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *encrypt/5*
  - *decrypt/5*
  - *encrypt_padded/5*
  - *decrypt_padded/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

encrypt/5

Encrypts block-aligned plaintext without padding using an explicit initialization vector.

Compilation flags:
`static`

Template:
`encrypt(Cipher,Key,IV,Plaintext,Ciphertext)`
Mode and number of proofs:
`encrypt(+object_identifier,+list(byte),+list(byte),+list(byte),--list(byte))` - `one_or_error`

Exceptions:

`Cipher` is a variable:

```
instantiation_error
```

`Cipher` is neither a variable nor a prepared-key block cipher object:

```
domain_error(block_cipher,Cipher)
```

`IV` is not a byte list of exactly one block:

```
type_error(list(byte,BlockSize),IV)
```

`Plaintext` is a variable or a partial list:

```
instantiation_error
```

`Plaintext` is neither a partial list nor a list:

```
type_error(list,Plaintext)
```

An element `Byte` of the `Plaintext` list is neither a variable nor an integer:

```
type_error(integer,Byte)
```

An element `Byte` of the `Plaintext` list is an integer but not a valid byte:

```
domain_error(byte,Byte)
```

`Plaintext` length is not block aligned:

```
domain_error(block_aligned_byte_length(BlockSize),Plaintext)
```

---

decrypt/5

Decrypts block-aligned ciphertext without removing padding using an explicit initialization vector.

Compilation flags:

```
static
```

Template:

```
decrypt(Cipher,Key,IV,Ciphertext,Plaintext)
```

Mode and number of proofs:

```
decrypt(+object_identifier,+list(byte),+list(byte),+list(byte),--list(byte)) -
one_or_error
```

Exceptions:

`Cipher` is a variable:

```
instantiation_error
```

`Cipher` is neither a variable nor a prepared-key block cipher object:

```
domain_error(block_cipher,Cipher)
```

`IV` is not a byte list of exactly one block:

```
type_error(list(byte,BlockSize),IV)
```

`Ciphertext` is a variable or a partial list:

```
instantiation_error
```

`Ciphertext` is neither a partial list nor a list:

```
type_error(list,Ciphertext)
```

An element `Byte` of the `Ciphertext` list is neither a variable nor an integer:

```
type_error(integer,Byte)
```

An element `Byte` of the `Ciphertext` list is an integer but not a valid byte:

```
domain_error(byte,Byte)
```

`Ciphertext` length is not block aligned:

```
domain_error(block_aligned_byte_length(BlockSize),Ciphertext)
```

---

encrypt_padded/5

Pads plaintext using PKCS#7 and encrypts it using an explicit initialization vector.

Compilation flags:

```
static
```

Template:

```
encrypt_padded(Cipher,Key,IV,Plaintext,Ciphertext)
```

Mode and number of proofs:

```
encrypt_padded(+object_identifier,+list(byte),+list(byte),+list(byte),
--list(byte)) - one_or_error
```

Exceptions:

`Cipher` is a variable:

```
instantiation_error
```

`Cipher` is neither a variable nor a prepared-key block cipher object:

```
domain_error(block_cipher,Cipher)
```

`Cipher` block size exceeds the PKCS#7 limit:

```
domain_error(pkcs7_block_size,BlockSize)
```

`IV` is not a byte list of exactly one block:

```
type_error(list(byte,BlockSize),IV)
```

`Plaintext` is a variable or a partial list:

```
instantiation_error
```

`Plaintext` is neither a partial list nor a list:

```
type_error(list,Plaintext)
```

An element `Byte` of the `Plaintext` list is neither a variable nor an integer:

```
type_error(integer,Byte)
```

An element `Byte` of the `Plaintext` list is an integer but not a valid byte:

```
domain_error(byte,Byte)
```

---

decrypt_padded/5

Decrypts ciphertext and validates and removes PKCS#7 padding using an explicit initialization vector.

Compilation flags:

```
static
```

Template:

```
decrypt_padded(Cipher,Key,IV,Ciphertext,Plaintext)
```

Mode and number of proofs:

```
decrypt_padded(+object_identifier,+list(byte),+list(byte),+list(byte),
--list(byte)) - one_or_error
```

Exceptions:

`Cipher` is a variable:

```
instantiation_error
```

`Cipher` is neither a variable nor a prepared-key block cipher object:

```
domain_error(block_cipher,Cipher)
```

`IV` is not a byte list of exactly one block:

```
type_error(list(byte,BlockSize),IV)
```

`Ciphertext` is a variable or a partial list:

```
instantiation_error
```

`Ciphertext` is neither a partial list nor a list:

```
type_error(list,Ciphertext)
```

An element `Byte` of the `Ciphertext` list is neither a variable nor an integer:

```
type_error(integer,Byte)
```

An element `Byte` of the `Ciphertext` list is an integer but not a valid byte:

```
domain_error(byte,Byte)
```

`Ciphertext` is empty:

```
domain_error(non_empty_ciphertext,Ciphertext)
```

`Ciphertext` length is not block aligned:

```
domain_error(block_aligned_byte_length(BlockSize),Ciphertext)
```

`Ciphertext` does not contain valid PKCS#7 padding:

```
domain_error(pkcs7_padding,Ciphertext)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.22.9 `ctr`

Generic Counter mode encryption and decryption using a whole-block big-endian counter.

Availability:

```
logtalk_load(block_ciphers(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *block_cipher_modes_common*

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *crypt/5*
  - *crypt/6*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

crypt/5

Encrypts or decrypts bytes using an explicit initial whole-block big-endian counter. The counter is incremented internally while processing the input but is not returned. The client is responsible for managing initial counters across calls and preventing reuse of a counter sequence with the same key.

Compilation flags:

`static`

Template:

`crypt(Cipher,Key,InitialCounter,Input,Output)`

Mode and number of proofs:

`crypt(+object_identifier,+list(byte),+list(byte),+list(byte),--list(byte))` - `one_or_error`

Exceptions:

`Cipher` is a variable:

`instantiation_error`

`Cipher` is neither a variable nor a prepared-key block cipher object:

`domain_error(block_cipher,Cipher)`

`InitialCounter` is not a byte list of exactly one block:

`type_error(list(byte,BlockSize),InitialCounter)`

`Input` is a variable or a partial list:

`instantiation_error`

`Input` is neither a partial list nor a list:

`type_error(list,Input)`

An element `Byte` of the `Input` list is neither a variable nor an integer:

`type_error(integer,Byte)`

An element `Byte` of the `Input` list is an integer but not a valid byte:

`domain_error(byte,Byte)`

crypt/6

Encrypts or decrypts bytes using an explicit initial whole-block big-endian counter and returns the next unused counter. The final counter equals the initial counter for empty input and is otherwise incremented once per processed input block, modulo the counter width.

Compilation flags:

`static`

Template:

`crypt(Cipher,Key,InitialCounter,Input,FinalCounter,Output)`

Mode and number of proofs:

`crypt(+object_identifier,+list(byte),+list(byte),+list(byte),--list(byte),--list(byte))` - `one_or_error`

Exceptions:

`Cipher` is a variable:

`instantiation_error`

`Cipher` is neither a variable nor a prepared-key block cipher object:

`domain_error(block_cipher,Cipher)`

`InitialCounter` is not a byte list of exactly one block:

`type_error(list(byte,BlockSize),InitialCounter)`

`Input` is a variable or a partial list:

`instantiation_error`

`Input` is neither a partial list nor a list:

`type_error(list,Input)`

An element `Byte` of the `Input` list is neither a variable nor an integer:

`type_error(integer,Byte)`

An element `Byte` of the `Input` list is an integer but not a valid byte:

`domain_error(byte,Byte)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.22.10 ecb

Generic Electronic Codebook mode encryption and decryption.

Availability:
`logtalk_load(block_ciphers(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *block_cipher_modes_common*
Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *encrypt/4*
  - *decrypt/4*
  - *encrypt_padded/4*
  - *decrypt_padded/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

encrypt/4

Encrypts block-aligned plaintext without padding.

Compilation flags:
`static`

Template:

encrypt(Cipher,Key,Plaintext,Ciphertext)

Mode and number of proofs:

encrypt(+object_identifier,+list(byte),+list(byte),--list(byte)) - one_or_error

Exceptions:

Cipher is a variable:

instantiation_error

Cipher is neither a variable nor a prepared-key block cipher object:

domain_error(block_cipher,Cipher)

Plaintext is a variable or a partial list:

instantiation_error

Plaintext is neither a partial list nor a list:

type_error(list,Plaintext)

An element Byte of the Plaintext list is neither a variable nor an integer:

type_error(integer,Byte)

An element Byte of the Plaintext list is an integer but not a valid byte:

domain_error(byte,Byte)

Plaintext length is not block aligned:

domain_error(block_aligned_byte_length(BlockSize),Plaintext)

---

decrypt/4

Decrypts block-aligned ciphertext without removing padding.

Compilation flags:

static

Template:

decrypt(Cipher,Key,Ciphertext,Plaintext)

Mode and number of proofs:

decrypt(+object_identifier,+list(byte),+list(byte),--list(byte)) - one_or_error

Exceptions:

Cipher is a variable:

instantiation_error

Cipher is neither a variable nor a prepared-key block cipher object:

domain_error(block_cipher,Cipher)

Ciphertext is a variable or a partial list:

instantiation_error

Ciphertext is neither a partial list nor a list:

type_error(list,Ciphertext)

An element Byte of the Ciphertext list is neither a variable nor an integer:

type_error(integer,Byte)

An element Byte of the Ciphertext list is an integer but not a valid byte:

domain_error(byte,Byte)

Ciphertext length is not block aligned:

domain_error(block_aligned_byte_length(BlockSize),Ciphertext)

encrypt_padded/4

Pads plaintext using PKCS#7 and encrypts it.

Compilation flags:
`static`

Template:
`encrypt_padded(Cipher,Key,Plaintext,Ciphertext)`
Mode and number of proofs:
`encrypt_padded(+object_identifier,+list(byte),+list(byte),--list(byte))` - `one_or_error`

Exceptions:
`Cipher` is a variable:
`instantiation_error`
`Cipher` is neither a variable nor a prepared-key block cipher object:
`domain_error(block_cipher,Cipher)`
`Plaintext` is a variable or a partial list:
`instantiation_error`
`Plaintext` is neither a partial list nor a list:
`type_error(list,Plaintext)`
An element `Byte` of the `Plaintext` list is neither a variable nor an integer:
`type_error(integer,Byte)`
An element `Byte` of the `Plaintext` list is an integer but not a valid byte:
`domain_error(byte,Byte)`
`Cipher` block size exceeds the PKCS#7 limit:
`domain_error(pkcs7_block_size,BlockSize)`

decrypt_padded/4

Decrypts ciphertext and validates and removes PKCS#7 padding.

Compilation flags:
`static`

Template:
`decrypt_padded(Cipher,Key,Ciphertext,Plaintext)`
Mode and number of proofs:
`decrypt_padded(+object_identifier,+list(byte),+list(byte),--list(byte))` - `one_or_error`

Exceptions:
`Cipher` is a variable:
`instantiation_error`

`Cipher` is neither a variable nor a prepared-key block cipher object:
`domain_error(block_cipher,Cipher)`
`Cipher` is a variable:
`instantiation_error`
`Cipher` is neither a variable nor a prepared-key block cipher object:
`domain_error(block_cipher,Cipher)`
`Ciphertext` is a variable or a partial list:
`instantiation_error`
`Ciphertext` is neither a partial list nor a list:
`type_error(list,Ciphertext)`
An element `Byte` of the `Ciphertext` list is neither a variable nor an integer:
`type_error(integer,Byte)`
An element `Byte` of the `Ciphertext` list is an integer but not a valid byte:
`domain_error(byte,Byte)`
`Ciphertext` is empty:
`domain_error(non_empty_ciphertext,Ciphertext)`
`Ciphertext` length is not block aligned:
`domain_error(block_aligned_byte_length(BlockSize),Ciphertext)`
`Ciphertext` does not contain valid PKCS#7 padding:
`domain_error(pkcs7_padding,Ciphertext)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.23 bm25_recommender

object

1.23.1 `bm25_recommender`

Content-based recommender using raw-count query profiles and Okapi BM25 relevance scores.

Availability:
`logtalk_load(bm25_recommender(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *recommender_common*

`public` *item_content_dataset_validation*

Uses:

list

type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_recommender/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_recommender/1* *recommend/4* *recommender_options/2* *score/4* *valid_option/1* *valid_options/1* *valid_recommender/1*

- *Public predicates*
  - *score_all/4*
  - *score_content/4*
  - *update_ratings/3*
  - *remove_ratings/3*
  - *extend_catalog/3*
  - *replace_content/3*
  - *remove_catalog/3*
  - *set_rating_scale/3*
- *Protected predicates*
  - *user_profile/5*
- *Private predicates*
  - *check_catalog_identifiers/1*
  - *check_rating_scale/2*
  - *check_scale_ratings/3*
  - *check_catalog_entries/1*
  - *check_existing_catalog_items/2*
  - *check_catalog_kind/3*
  - *check_new_catalog_items/2*
  - *rebuild_catalog_model/6*
  - *rating_removal_records/3*
  - *check_rating_updates/3*
  - *rebuild_feedback_model/9*
  - *updated_profiles/6*
  - *check_content_kind/3*
  - *score_catalog_items/6*
  - *score_query_items/4*
  - *catalog_score/6*
  - *content_documents/2*

    - *descriptor_document/2*
    - *count_length/3*
    - *profile_weights/3*
    - *check_finite_ratings/1*
  - *Operators*

Public predicates

score_all/4

Scores catalog items in input order, preserving duplicates and validating the model once. An empty batch validates the model and user.

Compilation flags:

`static`

Template:

`score_all(Recommender,User,Items,Scores)`

Mode and number of proofs:

`score_all(+compound,+atomic,+list(atomic),-list(pair))` - `one_or_error`

Exceptions:

A required argument or identifier is a variable:

`instantiation_error`

The model is invalid:

`domain_error(recommender,Recommender)`

A user or item identifier is not atomic:

`type_error(atomic,Identifier)`

Items is not a proper list:

`type_error(list,Items)`

An item is absent from the catalog:

`domain_error(catalog_item,Item)`

---

score_content/4

Scores supplied content of the trained representation using frozen corpus statistics. Unseen terms contribute to document length but not matching relevance.

Compilation flags:

`static`

Template:

`score_content(Recommender,User,Content,Score)`

Mode and number of proofs:

`score_content(+compound,+atomic,+compound,-float)` - `one_or_error`

Exceptions:

A required argument, feature, entry, or count is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

User is not atomic:

```
type_error(atomic,User)
```

A descriptor is invalid:

```
domain_error(item_content,Content)
```

The descriptor differs from the trained representation:

```
domain_error(content_representation,Content)
```

A content list is not a proper list:

```
type_error(list,List)
```

A vector entry is not a pair:

```
type_error(pair,Entry)
```

A vector repeats a feature key:

```
domain_error(duplicate_feature,Feature)
```

A vector count is not numeric:

```
type_error(number,Count)
```

A vector count is negative or nonfinite:

```
domain_error(non_negative_finite_weight,Count)
```

A nonzero vector count is not an integer:

```
type_error(integer,Count)
```

---

update_ratings/3

Returns a model with inserted or replaced ratings and rebuilt query profiles, retaining fitted corpus weights. New users are accepted but items must belong to the catalog.

Compilation flags:

```
static
```

Template:

```
update_ratings(Recommender,Ratings,UpdatedRecommender)
```

Mode and number of proofs:

```
update_ratings(+compound,+list(compound),-compound) - one_or_error
```

Exceptions:

A required argument, record, identifier, or value is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

Ratings is not a proper list:

```
type_error(list,Ratings)
```

An entry is not a rating record:

```
type_error(rating,Entry)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

A rating value is not numeric:

```
type_error(number,Rating)
```

A rating value is nonfinite:

```
domain_error(finite_rating,Rating)
```

A rating value is outside the stored scale:

```
domain_error(rating_scale(Min,Max),Rating)
```

An item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

A user-item pair occurs more than once in the updates:

```
domain_error(duplicate_rating,User-Item)
```

A selected rating is not positive in rating-weighted mode:

```
domain_error(positive_rating_weight,Rating)
```

---

remove_ratings/3

Returns a model with requested ratings removed and query profiles rebuilt. Missing and repeated pairs are accepted; no matching ratings returns the validated original model.

Compilation flags:

```
static
```

Template:

```
remove_ratings(Recommender,UserItemPairs,UpdatedRecommender)
```

Mode and number of proofs:

```
remove_ratings(+compound,+list(pair),-compound) - one_or_error
```

Exceptions:

A required argument, entry, or identifier is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

UserItemPairs is not a proper list:

```
type_error(list,UserItemPairs)
```

An entry is not a pair:

```
type_error(pair,Entry)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

No ratings would remain:

```
domain_error(non_empty_ratings,[])
```

A selected rating is not positive in rating-weighted mode:

```
domain_error(positive_rating_weight,Rating)
```

---

extend_catalog/3

Returns a model with new catalog items, refitting corpus statistics and all item weights. Descriptors must match the trained representation.

Compilation flags:

`static`

Template:

`extend_catalog(Recommender,ItemContents,UpdatedRecommender)`

Mode and number of proofs:

`extend_catalog(+compound,+list(pair),-compound)` - `one_or_error`

Exceptions:

A required argument, entry, identifier, feature, or count is a variable:

`instantiation_error`

The model is invalid:

`domain_error(recommender,Recommender)`

A supplied list is not proper:

`type_error(list,List)`

A catalog or vector entry is not a pair:

`type_error(pair,Entry)`

An item identifier is not atomic:

`type_error(atomic,Item)`

A batch repeats an item identifier:

`domain_error(duplicate_item,Item)`

An item already belongs to the catalog:

`domain_error(new_catalog_item,Item)`

A descriptor is invalid:

`domain_error(item_content,Content)`

A descriptor differs from the trained representation:

`domain_error(content_representation,Content)`

A vector repeats a feature key:

`domain_error(duplicate_feature,Feature)`

A vector count is not numeric:

`type_error(number,Count)`

A vector count is negative or nonfinite:

`domain_error(non_negative_finite_weight,Count)`

A nonzero vector count is not an integer:

`type_error(integer,Count)`

replace_content/3

Returns a model with replaced catalog descriptors, refitting corpus statistics, all item weights, and raw query profiles. Identifiers must already belong to the catalog.

Compilation flags:

```
static
```

Template:

```
replace_content(Recommender,ItemContents,UpdatedRecommender)
```

Mode and number of proofs:

```
replace_content(+compound,+list(pair),-compound) - one_or_error
```

Exceptions:

A required argument, entry, identifier, feature, or count is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

A supplied list is not proper:

```
type_error(list,List)
```

A catalog or vector entry is not a pair:

```
type_error(pair,Entry)
```

An item identifier is not atomic:

```
type_error(atomic,Item)
```

A batch repeats an item identifier:

```
domain_error(duplicate_item,Item)
```

An item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

A descriptor is invalid:

```
domain_error(item_content,Content)
```

A descriptor differs from the trained representation:

```
domain_error(content_representation,Content)
```

A vector repeats a feature key:

```
domain_error(duplicate_feature,Feature)
```

A vector count is not numeric:

```
type_error(number,Count)
```

A vector count is negative or nonfinite:

```
domain_error(non_negative_finite_weight,Count)
```

A nonzero vector count is not an integer:

```
type_error(integer,Count)
```

remove_catalog/3

Removes catalog items and their ratings, refitting remaining corpus weights and affected profiles. Missing and repeated identifiers are accepted.

Compilation flags:

`static`

Template:

`remove_catalog(Recommender,Items,UpdatedRecommender)`

Mode and number of proofs:

`remove_catalog(+compound,+list(atomic),-compound)` - `one_or_error`

Exceptions:

A required argument or identifier is a variable:

`instantiation_error`

The model is invalid:

`domain_error(recommender,Recommender)`

Items is not a proper list:

`type_error(list,Items)`

An item identifier is not atomic:

`type_error(atomic,Item)`

No catalog items would remain:

`domain_error(non_empty_catalog,[])`

No ratings would remain:

`domain_error(non_empty_ratings,[])`

A selected rating is not positive in rating-weighted mode:

`domain_error(positive_rating_weight,Rating)`

---

set_rating_scale/3

Changes only the stored rating scale to `none` or `scale(Min,Max)`, requiring finite ordered bounds containing all ratings. Ratings and relevance scores are not rescaled.

Compilation flags:

`static`

Template:

`set_rating_scale(Recommender,Scale,UpdatedRecommender)`

Mode and number of proofs:

`set_rating_scale(+compound,+term,-compound)` - `one_or_error`

Exceptions:

A required argument or scale bound is a variable:

`instantiation_error`

The model is invalid:

`domain_error(recommender,Recommender)`

A scale bound is not numeric:

```
type_error(number,Bound)
```

The scale descriptor is invalid:

```
domain_error(rating_scale,Scale)
```

Scale bounds are nonfinite or reversed:

```
domain_error(rating_scale,Min-Max)
```

A stored rating is outside the proposed scale:

```
domain_error(rating_scale(Min,Max),Rating)
```

Protected predicates

user_profile/5

Builds one raw-count query profile from the complete user history.

Compilation flags:

```
static
```

Template:

```
user_profile(User,Ratings,Documents,Options,Profile)
```

Mode and number of proofs:

```
user_profile(+atomic,+list(compound),+list(pair),+list(compound),-list(pair)) -
one_or_error
```

Exceptions:

A selected rating is not positive in rating-weighted mode:

```
domain_error(positive_rating_weight,Rating)
```

Private predicates

check_catalog_identifiers/1

Checks all catalog request identifiers before applying removals.

Compilation flags:

```
static
```

Template:

```
check_catalog_identifiers(Items)
```

Mode and number of proofs:

```
check_catalog_identifiers(+list(term)) - one_or_error
```

Exceptions:

An identifier is a variable:

```
instantiation_error
```

An identifier is not atomic:

`type_error(atomic,Item)`

---

check_rating_scale/2

Checks a proposed optional finite rating scale against existing ratings.

Compilation flags:
`static`

Template:
`check_rating_scale(Scale,Ratings)`
Mode and number of proofs:
`check_rating_scale(+term,+list(compound))` - `one_or_error`

Exceptions:
The scale or a bound is a variable:
`instantiation_error`
A bound is not numeric:
`type_error(number,Bound)`
The scale descriptor is invalid:
`domain_error(rating_scale,Scale)`
Bounds are nonfinite or reversed:
`domain_error(rating_scale,Min-Max)`
A rating is outside the proposed bounds:
`domain_error(rating_scale(Min,Max),Rating)`

---

check_scale_ratings/3

Checks stored ratings against inclusive validated scale bounds.

Compilation flags:
`static`

Template:
`check_scale_ratings(Ratings,Min,Max)`
Mode and number of proofs:
`check_scale_ratings(+list(compound),+number,+number)` - `one_or_error`

Exceptions:
A rating is outside the proposed bounds:
`domain_error(rating_scale(Min,Max),Rating)`

---

check_catalog_entries/1

Checks that all supplied catalog entries are pairs before canonicalization.

Compilation flags:
`static`

Template:
`check_catalog_entries(ItemContents)`
Mode and number of proofs:
`check_catalog_entries(+list(term))` - `one_or_error`

Exceptions:
An entry is a variable:
`instantiation_error`
An entry is not a pair:
`type_error(pair,Entry)`

---

check_existing_catalog_items/2

Checks that replacements use existing catalog identifiers.

Compilation flags:
`static`

Template:
`check_existing_catalog_items(Replacements,Contents)`
Mode and number of proofs:
`check_existing_catalog_items(+list(pair),+list(pair))` - `one_or_error`

Exceptions:
An item is absent from the catalog:
`domain_error(catalog_item,Item)`

---

check_catalog_kind/3

Checks the canonical catalog descriptor representation against the stored representation.

Compilation flags:
`static`

Template:
`check_catalog_kind(ItemContents,Kind,Diagnostics)`
Mode and number of proofs:
`check_catalog_kind(+list(pair),+atom,+list(compound))` - `one_or_error`

Exceptions:

A descriptor differs from the trained representation:

```
domain_error(content_representation,Content)
```

---

check_new_catalog_items/2

Checks that additions use new catalog identifiers.

Compilation flags:

```
static
```

Template:

```
check_new_catalog_items(Additions,Contents)
```

Mode and number of proofs:

```
check_new_catalog_items(+list(pair),+list(pair)) - one_or_error
```

Exceptions:

An item already belongs to the catalog:

```
domain_error(new_catalog_item,Item)
```

---

rebuild_catalog_model/6

Refits corpus statistics and all item weights using prepared raw-count profiles.

Compilation flags:

```
static
```

Template:

```
rebuild_catalog_model(Ratings,Contents,Profiles,Scale,Diagnostics,
UpdatedRecommender)
```

Mode and number of proofs:

```
rebuild_catalog_model(+list(compound),+list(pair),+list(pair),+term,
+list(compound),-compound) - one_or_error
```

Exceptions:

A nonzero vector count is not an integer:

```
type_error(integer,Count)
```

---

rating_removal_records/3

Validates user-item pairs and collects matching stored ratings, ignoring missing pairs.

Compilation flags:

```
static
```

Template:

```
rating_removal_records(UserItemPairs,Ratings,Removals)
```

Mode and number of proofs:

```
rating_removal_records(+list(pair),+list(compound),-list(compound)) - one_or_error
```

Exceptions:

An entry or identifier is a variable:

```
instantiation_error
```

An entry is not a pair:

```
type_error(pair,Entry)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

---

check_rating_updates/3

Checks rating records against the catalog and stored rating scale.

Compilation flags:

```
static
```

Template:

```
check_rating_updates(Ratings,Contents,Scale)
```

Mode and number of proofs:

```
check_rating_updates(+list(compound),+list(pair),+term) - one_or_error
```

Exceptions:

A record, identifier, or value is a variable:

```
instantiation_error
```

An entry is not a rating record:

```
type_error(rating,Entry)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

A rating value is not numeric:

```
type_error(number,Rating)
```

A rating value is nonfinite:

```
domain_error(finite_rating,Rating)
```

A rating value is outside the stored scale:

```
domain_error(rating_scale(Min,Max),Rating)
```

An item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

rebuild_feedback_model/9

Rebuilds affected raw-count profiles and diagnostics, retaining fitted corpus and item weights.

Compilation flags:

```
static
```

Template:

```
rebuild_feedback_model(Ratings,Contents,ItemWeights,Profiles,Corpus,Scale,
Diagnostics,AffectedUsers,UpdatedRecommender)
```

Mode and number of proofs:

```
rebuild_feedback_model(+list(compound),+list(pair),+list(pair),+list(pair),
+compound,+term,+list(compound),+list(atomic),-compound) - one_or_error
```

Exceptions:

A nonzero vector count is not an integer:

```
type_error(integer,Count)
```

A selected rating is not positive in rating-weighted mode:

```
domain_error(positive_rating_weight,Rating)
```

---

updated_profiles/6

Rebuilds affected users from complete history and retains unaffected raw profiles.

Compilation flags:

```
static
```

Template:

```
updated_profiles(Ratings,Contents,Options,Profiles,AffectedUsers,UpdatedProfiles)
```

Mode and number of proofs:

```
updated_profiles(+list(compound),+list(pair),+list(compound),+list(pair),
+list(atomic),-list(pair)) - one_or_error
```

Exceptions:

A nonzero vector count is not an integer:

```
type_error(integer,Count)
```

A selected rating is not positive in rating-weighted mode:

```
domain_error(positive_rating_weight,Rating)
```

---

check_content_kind/3

Checks that supplied content matches the stored representation.

Compilation flags:
`static`

Template:
`check_content_kind(Kind,Content,Diagnostics)`
Mode and number of proofs:
`check_content_kind(+atom,+compound,+list(compound))` - `one_or_error`

Exceptions:
The descriptor differs from the trained representation:
`domain_error(content_representation,Content)`

---

score_catalog_items/6

Validates and scores catalog identifiers in input order using validated model data.

Compilation flags:
`static`

Template:
`score_catalog_items(ItemWeights,Profiles,User,Items,Scores,Options)`
Mode and number of proofs:
`score_catalog_items(+list(pair),+list(pair),+atomic,+list(atomic),-list(pair),+list(compound))` - `one_or_error`

Exceptions:
An identifier is a variable:
`instantiation_error`
An identifier is not atomic:
`type_error(atomic,Item)`
An item is absent from the catalog:
`domain_error(catalog_item,Item)`

---

score_query_items/4

Restores requested score order, checking identifier errors in original request order.

Compilation flags:
`static`

Template:

```
score_query_items(ResolvedScores,Query,Items,Scores)
```

Mode and number of proofs:

```
score_query_items(+list(pair),+list(pair),+list(atomic),-list(pair)) -
one_or_error
```

Exceptions:

An identifier is a variable:

```
instantiation_error
```

An identifier is not atomic:

```
type_error(atomic,Item)
```

An item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

---

catalog_score/6

Scores a catalog identifier using validated model data.

Compilation flags:

```
static
```

Template:

```
catalog_score(ItemWeights,Profiles,User,Item,Score,Options)
```

Mode and number of proofs:

```
catalog_score(+list(pair),+list(pair),+atomic,+atomic,-float,+list(compound)) -
one_or_error
```

Exceptions:

An item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

---

content_documents/2

Converts canonical descriptors to raw counts and complete document lengths.

Compilation flags:

```
static
```

Template:

```
content_documents(Contents,Documents)
```

Mode and number of proofs:

```
content_documents(+list(pair),-list(pair)) - one_or_error
```

Exceptions:

A nonzero vector count is not an integer:

```
type_error(integer,Count)
```

---

descriptor_document/2

Converts a canonical descriptor to its raw-count document.

Compilation flags:
`static`

Template:
`descriptor_document(Content,Document)`
Mode and number of proofs:
`descriptor_document(+compound,-compound)` - `one_or_error`

Exceptions:
A nonzero vector count is not an integer:
`type_error(integer,Count)`

---

count_length/3

Checks canonical nonzero count values and sums the document length.

Compilation flags:
`static`

Template:
`count_length(Counts,Length0,Length)`
Mode and number of proofs:
`count_length(+list(pair),+integer,-integer)` - `one_or_error`

Exceptions:
A nonzero vector count is not an integer:
`type_error(integer,Count)`

---

profile_weights/3

Assigns uniform or strictly positive rating weights to selected count vectors.

Compilation flags:
`static`

Template:
`profile_weights(Selected,Weighting,Weighted)`
Mode and number of proofs:
`profile_weights(+list(pair),+atom,-list(pair))` - `one_or_error`

Exceptions:

A selected rating is not positive in rating-weighted mode:
`domain_error(positive_rating_weight,Rating)`

---

check_finite_ratings/1

Checks that rating values are finite.

Compilation flags:
`static`

Template:
`check_finite_ratings(Ratings)`
Mode and number of proofs:
`check_finite_ratings(+list(compound))` - `one_or_error`

Exceptions:
A rating value is nonfinite:
`domain_error(finite_rating,Rating)`

Operators

(none)

See also

recommender_protocol, *item_content_dataset_protocol*, *tfidf_recommender*

1.24 borda_ranker

object

1.24.1 borda_ranker

Borda grouped-ranking ranker. Learns one deterministic score per item from a dataset object implementing the `ranking_dataset_protocol` protocol by summing per-group Borda points and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:
`logtalk_load(borda_ranker(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public ranking_dataset_common
public score_ranker_model_common
```

Uses:

avltree

Remarks:

(none)

Inherited public predicates:

check_option/1 check_options/1 check_ranker/1 default_option/1 default_options/1 diagnostic/2 diagnostics/2 export_to_clauses/4 export_to_file/4 grouped_dataset_groups/2 grouped_dataset_items/2 grouped_dataset_items/3 grouped_dataset_summary/2 learn/2 learn/3 multiplayer_dataset_connected_components/2 multiplayer_dataset_declared_items/2 multiplayer_dataset_declared_matches/2 multiplayer_dataset_events/2 multiplayer_dataset_items/2 multiplayer_dataset_match_teams/3 multiplayer_dataset_matches/2 multiplayer_dataset_summary/2 option/2 option/3 pairwise_dataset_connected_components/2 pairwise_dataset_declared_items/2 pairwise_dataset_items/2 pairwise_dataset_matchups/2 pairwise_dataset_preferences/2 pairwise_dataset_summary/2 pairwise_dataset_win_totals/2 pairwise_measurement_dataset_connected_components/2 pairwise_measurement_dataset_declared_items/2 pairwise_measurement_dataset_items/2 pairwise_measurement_dataset_measurements/2 pairwise_measurement_dataset_summary/2 print_ranker/1 rank/3 ranker_options/2 scores/2 temporal_pairwise_dataset_connected_components/2 temporal_pairwise_dataset_declared_items/2 temporal_pairwise_dataset_games/2 temporal_pairwise_dataset_games/3 temporal_pairwise_dataset_items/2 temporal_pairwise_dataset_periods/2 temporal_pairwise_dataset_summary/2 valid_option/1 valid_options/1 valid_ranker/1 validate_grouped_dataset/1 validate_grouped_dataset/2 validate_multiplayer_dataset/1 validate_multiplayer_dataset/2 validate_pairwise_dataset/1 validate_pairwise_dataset/2 validate_pairwise_measurement_dataset/1 validate_pairwise_measurement_dataset/2 validate_temporal_pairwise_dataset/1 validate_temporal_pairwise_dataset/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ranking_dataset_protocol, *ranker_protocol*, *copeland_ranker*

1.25 bradley_terry_ranker

object

1.25.1 `bradley_terry_ranker`

Bradley-Terry pairwise preference ranker. Learns one positive strength parameter per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol when the directed win graph admits a finite Bradley-Terry maximum-likelihood estimate, and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:
`logtalk_load(bradley_terry_ranker(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ranking_dataset_common*
`public` *pairwise_strength_ranker_common*
`public` *score_ranker_common*

Uses:
avltree
format
list

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranking_dataset_protocol*, *ranker_protocol*, *regularized_bradley_terry_ranker*

1.26 bson

object

1.26.1 `bson`

BSON format exporter and importer using atoms for decoded strings and keys.

Availability:
`logtalk_load(bson(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-02

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *bson(atom)*

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.26.2 `bson(StringRepresentation)`

- `StringRepresentation` - Text representation to be used when decoding BSON strings and keys. Possible values are `atom` (default), `chars`, and `codes`.

BSON format exporter and importer.

Availability:
`logtalk_load(bson(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Uses:
byte_order
ieee_754(Precision,ByteOrder,NaNRepresentation)
list
utf_8

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a list of bytes in the BSON format returning the corresponding document term. Throws an error when parsing is not possible.

Compilation flags:
`static`

Template:
`parse(Bytes,Document)`
Mode and number of proofs:
`parse(@list(byte),-ground)` - `one_or_error`

Exceptions:
`Bytes` is not a ground term:
`instantiation_error`
`Bytes` is ground but not a valid BSON byte sequence:
`domain_error(bson_byte_sequence,Bytes)`

---

generate/2

Generates a list of bytes in the BSON format representing the given document term. Throws an error when generating is not possible.

Compilation flags:
`static`

Template:
`generate(Document,Bytes)`
Mode and number of proofs:
`generate(@ground,-list(byte))` - `one_or_error`

Exceptions:
`Document` is a variable:
`instantiation_error`
`Document` has no corresponding BSON representation:
`domain_error(bson_term,Document)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.27 byte_order

object

1.27.1 `byte_order`

Predicates for converting between integers and fixed-size byte lists in big-endian and little-endian order.

Availability:
`logtalk_load(byte_order(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-17

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *integer_to_bytes/4*
  - *integer_to_bytes/5*
  - *bytes_to_integer/3*
  - *bytes_to_integer/5*
  - *signed_integer_to_bytes/4*
  - *signed_integer_to_bytes/5*
  - *bytes_to_signed_integer/3*
  - *bytes_to_signed_integer/5*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

integer_to_bytes/4

Converts the low-order bytes of an integer into a byte list with the requested byte order and size.

Compilation flags:
`static`

Template:
`integer_to_bytes(Order,ByteCount,Integer,Bytes)`
Mode and number of proofs:
`integer_to_bytes(+atom,+integer,+integer,-list(integer))` - `one`

---

integer_to_bytes/5

Converts the low-order bytes of an integer into a byte list with the requested byte order and size, represented as a difference list.

Compilation flags:
`static`

Template:
`integer_to_bytes(Order,ByteCount,Integer,Bytes,Tail)`
Mode and number of proofs:
`integer_to_bytes(+atom,+integer,+integer,-list(integer),--term)` - `one`

---

bytes_to_integer/3

Converts a byte list with the requested byte order into an unsigned integer.

Compilation flags:
`static`

Template:
`bytes_to_integer(Order,Bytes,Integer)`
Mode and number of proofs:
`bytes_to_integer(+atom,+list(integer),-integer)` - `one`

---

bytes_to_integer/5

Consumes the requested number of bytes from a list and converts them with the requested byte order into an unsigned integer.

Compilation flags:
`static`

Template:
`bytes_to_integer(Order,ByteCount,Bytes,Integer,Rest)`
Mode and number of proofs:
`bytes_to_integer(+atom,+integer,+list(integer),-integer,-list(integer))` - `one`

---

signed_integer_to_bytes/4

Converts the low-order bytes of a signed integer two's-complement representation into a byte list with the requested byte order and size.

Compilation flags:
`static`

Template:
`signed_integer_to_bytes(Order,ByteCount,Integer,Bytes)`
Mode and number of proofs:
`signed_integer_to_bytes(+atom,+integer,+integer,-list(integer))` - `one`

---

signed_integer_to_bytes/5

Converts the low-order bytes of a signed integer two's-complement representation into a byte list with the requested byte order and size, represented as a difference list.

Compilation flags:
`static`

Template:
`signed_integer_to_bytes(Order,ByteCount,Integer,Bytes,Tail)`
Mode and number of proofs:
`signed_integer_to_bytes(+atom,+integer,+integer,-list(integer),--term)` - `one`

---

bytes_to_signed_integer/3

Converts a byte list with the requested byte order from a two's-complement representation into a signed integer.

Compilation flags:

```
static
```

Template:

```
bytes_to_signed_integer(Order,Bytes,Integer)
```

Mode and number of proofs:

```
bytes_to_signed_integer(+atom,+list(integer),-integer) - one
```

---

bytes_to_signed_integer/5

Consumes the requested number of bytes from a list and converts them with the requested byte order from a two's-complement representation into a signed integer.

Compilation flags:

```
static
```

Template:

```
bytes_to_signed_integer(Order,ByteCount,Bytes,Integer,Rest)
```

Mode and number of proofs:

```
bytes_to_signed_integer(+atom,+integer,+list(integer),-integer,-list(integer)) -
one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.28 c45_classifier

object

1.28.1 c45_classifier

C4.5 decision tree learning algorithm. Builds a decision tree from a dataset object implementing the `dataset_protocol` protocol and provides predicates for exporting the learned tree as a list of predicate clauses or to a file. Supports both discrete and continuous attributes, handles missing values, and supports tree pruning.

Availability:

```
logtalk_load(c45_classifier(loader))
```

Author: Paulo Moura
Version: 3:0:0
Date: 2026-10-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *classifier_common*

Uses:

format
list
numberlist
pairs
type

Remarks:

(none)

Inherited public predicates:

check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *prune/5*
  - *prune/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

prune/5

Prunes a decision tree using pessimistic error pruning (PEP). This post-pruning method estimates error rates using the upper confidence bound of the binomial distribution with the given confidence factor and replaces subtrees with leaf nodes when doing so would not increase the estimated error. Pruning helps reduce overfitting and can improve generalization to unseen data.

Compilation flags:

```
static
```

Template:

```
prune(Dataset,Tree,ConfidenceFactor,MinInstances,PrunedTree)
```

Mode and number of proofs:

```
prune(+object_identifier,+classifier,+float,+positive_integer,-classifier) - one
```

Exceptions:

An input parameter or tree is a variable:

```
instantiation_error
```

Confidence factor is not a float in (0.0, 0.5]:

```
domain_error(confidence_factor,ConfidenceFactor)
```

Minimum instances is not a positive integer:

```
domain_error(positive_integer,MinInstances)
```

Tree is invalid:

```
domain_error(classifier,Tree)
```

Dataset is empty:

```
domain_error(non_empty_dataset,Dataset)
```

Dataset class values are invalid:

```
domain_error(class_values,ClassValues)
```

Dataset attribute declarations are invalid:

```
domain_error(attribute_declarations,Attribute)
```

A required attribute binding is missing:

```
existence_error(attribute,Attribute)
```

Remarks:

- Confidence factor: The confidence factor must be a float in `(0.0, 0.5]` so that the error estimate is an upper confidence bound. Lower values increase the estimated uncertainty. The default value is `0.25`.
- Minimum node population: A non-empty node with weighted population below this positive integer is replaced by its majority leaf. This does not require every leaf to meet the minimum. The default value is `2`.
- Statistical basis: The pruning uses the upper confidence bound of the binomial distribution to estimate the true error rate.

prune/3

Prunes a decision tree using pessimistic error pruning (PEP) with default parameter values. Calls `prune/5` with `ConfidenceFactor = 0.25` and `MinInstances = 2`.

Compilation flags:

`static`

Template:

`prune(Dataset,Tree,PrunedTree)`

Mode and number of proofs:

`prune(+object_identifier,+classifier,-classifier)` - `one`

Exceptions:

Tree is a variable:

`instantiation_error`

Tree is invalid:

`domain_error(classifier,Tree)`

Dataset is empty:

`domain_error(non_empty_dataset,Dataset)`

Dataset class values are invalid:

`domain_error(class_values,ClassValues)`

Dataset attribute declarations are invalid:

`domain_error(attribute_declarations,Attribute)`

A required attribute binding is missing:

`existence_error(attribute,Attribute)`

Remarks:

- Default parameters: Uses a confidence factor of `0.25` and minimum weighted node population of `2`. Existing leaves and empty branches are preserved.

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *isolation_forest_anomaly_detector*, *knn_classifier*, *naive_bayes_classifier*, *nearest_centroid_classifier*, *random_forest_classifier*, *adaptive_boosting_classifier*

1.29 cartesian_products

object

1.29.1 `cartesian_products`

Implementation of Cartesian-product operations over lists.

Availability:
`logtalk_load(cartesian_products(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-05-21

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *cartesian_products_protocol*
Uses:
fast_random(Algorithm)
list

Remarks:
(none)

Inherited public predicates:
cartesian_product/2 *cartesian_product/3* *cartesian_tuple/2* *cartesian_tuple/3* *cartesian_tuple_index/3* *cartesian_tuple_index/4* *count_cartesian_product/2* *count_distinct_cartesian_product/2* *distinct_cartesian_product/2* *distinct_cartesian_product/3* *distinct_cartesian_tuple/2* *distinct_cartesian_tuple/3* *distinct_cartesian_tuple_index/3* *next_cartesian_tuple/3* *nth_cartesian_tuple/3* *nth_cartesian_tuple/4* *nth_distinct_cartesian_tuple/3* *previous_cartesian_tuple/3* *random_cartesian_tuple/2* *random_distinct_cartesian_tuple/2* *sample_cartesian_tuples/3* *sample_distinct_cartesian_tuples/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.29.2 cartesian_products_protocol

Protocol for Cartesian-product operations over lists.

Availability:
`logtalk_load(cartesian_products(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-12

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *cartesian_product/2*
  - *cartesian_tuple/2*
  - *cartesian_product/3*
  - *cartesian_tuple/3*
  - *distinct_cartesian_product/2*
  - *distinct_cartesian_tuple/2*
  - *distinct_cartesian_product/3*
  - *distinct_cartesian_tuple/3*

- count_cartesian_product/2
- count_distinct_cartesian_product/2
- nth_cartesian_tuple/3
- nth_cartesian_tuple/4
- cartesian_tuple_index/3
- cartesian_tuple_index/4
- nth_distinct_cartesian_tuple/3
- distinct_cartesian_tuple_index/3
- random_cartesian_tuple/2
- sample_cartesian_tuples/3
- random_distinct_cartesian_tuple/2
- sample_distinct_cartesian_tuples/3
- next_cartesian_tuple/3
- previous_cartesian_tuple/3

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

cartesian_product/2

Generates the Cartesian product of a list of lists using default order.

Compilation flags:
```
static
```

Template:
```
cartesian_product(Lists,Product)
```
Mode and number of proofs:
```
cartesian_product(+list(list),-list) - one
```

---

cartesian_tuple/2

True iff the second argument is a tuple obtained by selecting one element from each list in the first argument using default order.

Compilation flags:
```
static
```

Template:
```
cartesian_tuple(Lists,Tuple)
```
Mode and number of proofs:
```
cartesian_tuple(+list(list),-list) - one_or_more
```

---

cartesian_product/3

Generates the Cartesian product with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`cartesian_product(Lists,Order,Product)`
Mode and number of proofs:
`cartesian_product(+list(list),+atom,-list)` - `one`

---

cartesian_tuple/3

True iff the third argument is a Cartesian-product tuple with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`cartesian_tuple(Lists,Order,Tuple)`
Mode and number of proofs:
`cartesian_tuple(+list(list),+atom,-list)` - `one_or_more`

---

distinct_cartesian_product/2

Generates the Cartesian product after deduplicating repeated values in each factor list using default order.

Compilation flags:
`static`

Template:
`distinct_cartesian_product(Lists,Product)`
Mode and number of proofs:
`distinct_cartesian_product(+list(list),-list)` - `one`

---

distinct_cartesian_tuple/2

True iff the second argument is a tuple in the distinct Cartesian product of the factor lists using default order.

Compilation flags:
`static`

Template:
`distinct_cartesian_tuple(Lists,Tuple)`
Mode and number of proofs:
`distinct_cartesian_tuple(+list(list),-list)` - `one_or_more`

---

distinct_cartesian_product/3

Generates the distinct Cartesian product with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_cartesian_product(Lists,Order,Product)`
Mode and number of proofs:
`distinct_cartesian_product(+list(list),+atom,-list)` - `one`

---

distinct_cartesian_tuple/3

True iff the third argument is a tuple in the distinct Cartesian product with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_cartesian_tuple(Lists,Order,Tuple)`
Mode and number of proofs:
`distinct_cartesian_tuple(+list(list),+atom,-list)` - `one_or_more`

---

count_cartesian_product/2

Counts the number of tuples in the Cartesian product of a list of lists.

Compilation flags:
```
static
```

Template:
```
count_cartesian_product(Lists,Count)
```
Mode and number of proofs:
```
count_cartesian_product(+list(list),-integer) - one
```

---

count_distinct_cartesian_product/2

Counts the number of tuples in the distinct Cartesian product after deduplicating repeated values in each factor list.

Compilation flags:
```
static
```

Template:
```
count_distinct_cartesian_product(Lists,Count)
```
Mode and number of proofs:
```
count_distinct_cartesian_product(+list(list),-integer) - one
```

---

nth_cartesian_tuple/3

Returns the Cartesian-product tuple at a given zero-based index using default order.

Compilation flags:
```
static
```

Template:
```
nth_cartesian_tuple(Lists,Index,Tuple)
```
Mode and number of proofs:
```
nth_cartesian_tuple(+list(list),+integer,-list) - zero_or_one
```

---

nth_cartesian_tuple/4

Returns the Cartesian-product tuple at a given zero-based index in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`nth_cartesian_tuple(Lists,Order,Index,Tuple)`
Mode and number of proofs:
`nth_cartesian_tuple(+list(list),+atom,+integer,-list)` - `zero_or_one`

---

cartesian_tuple_index/3

Returns the zero-based index of a Cartesian-product tuple using default order.

Compilation flags:
`static`

Template:
`cartesian_tuple_index(Lists,Tuple,Index)`
Mode and number of proofs:
`cartesian_tuple_index(+list(list),+list,-integer)` - `zero_or_one`

---

cartesian_tuple_index/4

Returns the zero-based index of a Cartesian-product tuple in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`cartesian_tuple_index(Lists,Order,Tuple,Index)`
Mode and number of proofs:
`cartesian_tuple_index(+list(list),+atom,+list,-integer)` - `zero_or_one`

---

nth_distinct_cartesian_tuple/3

Returns the distinct Cartesian-product tuple at a given zero-based index in default generation order.

Compilation flags:
`static`

Template:
`nth_distinct_cartesian_tuple(Lists,Index,Tuple)`
Mode and number of proofs:
`nth_distinct_cartesian_tuple(+list(list),+integer,-list)` - `zero_or_one`

---

distinct_cartesian_tuple_index/3

Returns the zero-based index of a distinct Cartesian-product tuple in default generation order.

Compilation flags:
`static`

Template:
`distinct_cartesian_tuple_index(Lists,Tuple,Index)`
Mode and number of proofs:
`distinct_cartesian_tuple_index(+list(list),+list,-integer)` - `zero_or_one`

---

random_cartesian_tuple/2

Returns a random Cartesian-product tuple.

Compilation flags:
`static`

Template:
`random_cartesian_tuple(Lists,Tuple)`
Mode and number of proofs:
`random_cartesian_tuple(+list(list),-list)` - `zero_or_one`

---

sample_cartesian_tuples/3

Returns SampleCount random Cartesian-product tuples, sampled with replacement.

Compilation flags:
`static`

Template:
`sample_cartesian_tuples(Lists,SampleCount,Samples)`
Mode and number of proofs:
`sample_cartesian_tuples(+list(list),+integer,-list)` - `zero_or_one`

---

random_distinct_cartesian_tuple/2

Returns a random tuple from the distinct Cartesian product after deduplicating repeated values in each factor list.

Compilation flags:
`static`

Template:
`random_distinct_cartesian_tuple(Lists,Tuple)`
Mode and number of proofs:
`random_distinct_cartesian_tuple(+list(list),-list)` - `zero_or_one`

---

sample_distinct_cartesian_tuples/3

Returns SampleCount random tuples from the distinct Cartesian product, sampled with replacement.

Compilation flags:
`static`

Template:
`sample_distinct_cartesian_tuples(Lists,SampleCount,Samples)`
Mode and number of proofs:
`sample_distinct_cartesian_tuples(+list(list),+integer,-list)` - `zero_or_one`

---

next_cartesian_tuple/3

Returns the next distinct Cartesian-product tuple in lexicographic order induced by the factor lists.

Compilation flags:
`static`

Template:
`next_cartesian_tuple(Lists,Tuple,Next)`
Mode and number of proofs:
`next_cartesian_tuple(+list(list),+list,-list)` - `zero_or_one`

---

previous_cartesian_tuple/3

Returns the previous distinct Cartesian-product tuple in lexicographic order induced by the factor lists.

Compilation flags:
`static`

Template:
`previous_cartesian_tuple(Lists,Tuple,Previous)`
Mode and number of proofs:
`previous_cartesian_tuple(+list(list),+list,-list)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.30 cbor

object

1.30.1 cbor

Concise Binary Object Representation (CBOR) format exporter and importer. Uses atoms to represent decoded CBOR strings.

Availability:
`logtalk_load(cbor(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2021-03-04

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *cbor(atom)*

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.30.2 cbor(StringRepresentation)

- `StringRepresentation` - Text representation to be used when decoding CBOR strings. Possible values are `atom` (default), `chars`, and `codes`.

Concise Binary Object Representation (CBOR) format exporter and importer.

Availability:
`logtalk_load(cbor(loader))`

Author: Paulo Moura
Version: 0:12:3
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Uses:
ieee_754(Precision,ByteOrder,NaNRepresentation)
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a list of bytes in the CBOR format returning the corresponding term representation. Throws an error when parsing is not possible (usually due to an invalid byte sequence).

Compilation flags:
`static`

Template:
`parse(Bytes,Term)`
Mode and number of proofs:
`parse(@list(byte),-ground)` - `one_or_error`

Exceptions:

`Bytes` is not a ground term:
`instantiation_error`
`Bytes` is not a valid CBOR byte sequence:
`domain_error(cbor_byte_sequence,Bytes)`

`generate/2`

Generates a list of bytes in the CBOR format representing the given term. Throws an error when generating is not possible (usually due to a term that have no CBOR corresponding representation).

Compilation flags:
`static`

Template:
`generate(Term,Bytes)`
Mode and number of proofs:
`generate(@ground,-list(byte))` - `one_or_error`

Exceptions:
`Term` is a variable:
`instantiation_error`
`Term` is neither a variable nor has a corresponding CBOR representation:
`domain_error(cbor_term,Term)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.31 ccsds_frames

object

1.31.1 ccsds_aos_frames(FrameLength,InsertZoneLength,HasOCF,HasFECF)

- `FrameLength` - Fixed AOS transfer frame length in octets.
- `InsertZoneLength` - Length in octets of the optional insert zone when present.
- `HasOCF` - Boolean indicating if frames include the operational control field.
- `HasFECF` - Boolean indicating if frames include the frame error control field.

CCSDS advanced orbiting systems transfer frame parser and generator.

Availability:
`logtalk_load(ccsds_frames(loader))`

Author: Paulo Moura
Version: 1:0:2
Date: 2026-06-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ccsds_frame_protocol*
Uses:
list
reader
type

Remarks:
(none)

Inherited public predicates:
data_field/2 *fecf/2* *frame_type/2* *generate/2* *generate/3* *ocf/2* *parse/2* *spacecraft_id/2* *valid/1* *version/2* *virtual_channel_id/2*

- *Public predicates*
  - *virtual_channel_frame_count/2*
  - *signaling_field/2*
  - *insert_zone/2*
  - *update_fecf/2*
  - *verify_fecf/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

virtual_channel_frame_count/2

Extracts the virtual channel frame count.

Compilation flags:
```
static
```

Template:
```
virtual_channel_frame_count(Frame,Count)
```
Mode and number of proofs:
```
virtual_channel_frame_count(+compound,-integer) - one
```

---

signaling_field/2

Extracts the signaling field term.

Compilation flags:
```
static
```

Template:
```
signaling_field(Frame,SignalingField)
```
Mode and number of proofs:
```
signaling_field(+compound,-compound) - one
```

---

insert_zone/2

Extracts the optional insert zone. Returns `none` when absent.

Compilation flags:
```
static
```

Template:
```
insert_zone(Frame,InsertZone)
```
Mode and number of proofs:
```
insert_zone(+compound,-compound) - one
```

---

update_fecf/2

Computes the correct AOS transfer frame FECF for the selected object configuration and returns the corresponding updated frame term.

Compilation flags:
`static`

Template:
`update_fecf(Frame,UpdatedFrame)`
Mode and number of proofs:
`update_fecf(+compound,-compound)` - `one_or_error`

Exceptions:
`Frame` is a variable:
`instantiation_error`
`Frame` is neither a variable nor a valid AOS transfer frame term:
`domain_error(ccsds_frame_term,Frame)`

---

verify_fecf/1

True if the AOS transfer frame term contains the correct FECF for the selected object configuration.

Compilation flags:
`static`

Template:
`verify_fecf(Frame)`
Mode and number of proofs:
`verify_fecf(@compound)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.31.2 ccsds_frame_protocol

CCSDS transfer frame parser and generator protocol.

Availability:
`logtalk_load(ccsds_frames(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
  - *generate/3*
  - *valid/1*
  - *update_fecf/2*
  - *verify_fecf/1*
  - *frame_type/2*
  - *version/2*
  - *spacecraft_id/2*
  - *virtual_channel_id/2*
  - *data_field/2*
  - *ocf/2*
  - *fecf/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses CCSDS transfer frame(s) from a source. Supported source terms are `file(File)`, `stream(Stream)`, and `bytes(Bytes)`.

Compilation flags:
`static`

Template:
`parse(Source,Frames)`
Mode and number of proofs:
`parse(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is not a valid CCSDS frame source term:
`domain_error(ccsds_frame_source,Source)`
`Source` bytes are not a valid CCSDS frame byte sequence for the selected frame type:
`domain_error(ccsds_frame_byte_sequence,Bytes)`

---

generate/2

Generates CCSDS transfer frame bytes to a sink. Supported sink terms are `file(File)`, `stream(Stream)`, and `bytes(Bytes)`.

Compilation flags:
`static`

Template:
`generate(Sink,Frames)`
Mode and number of proofs:
`generate(+compound,+list(compound))` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is not a valid CCSDS frame sink term:
`domain_error(ccsds_frame_sink,Sink)`
`Frames` is not a valid list of CCSDS transfer frame terms for the selected frame type:
`domain_error(ccsds_frame_term,Frame)`

---

generate/3

Generates a list of bytes from a single CCSDS transfer frame term with an open tail. When the selected object includes an FECF, the generated bytes use a freshly computed FECF derived from the remaining frame fields.

Compilation flags:
`static`

Template:
`generate(Frame,Bytes,Tail)`
Mode and number of proofs:
`generate(+compound,-list(byte),--variable)` - `one_or_error`

Exceptions:
`Frame` is a variable:
`instantiation_error`
`Frame` is not a valid CCSDS transfer frame term for the selected frame type:
`domain_error(ccsds_frame_term,Frame)`

---

valid/1

True if the argument is a valid transfer frame term for the selected object and, when applicable, carries the correct FECF for the remaining frame fields.

Compilation flags:
`static`

Template:
`valid(Frame)`
Mode and number of proofs:
`valid(@compound)` - `zero_or_one`

---

update_fecf/2

Computes the correct frame error control field for the selected object configuration and returns the corresponding updated frame term.

Compilation flags:
`static`

Template:
`update_fecf(Frame,UpdatedFrame)`
Mode and number of proofs:
`update_fecf(+compound,-compound)` - `one_or_error`

Exceptions:

`Frame` is a variable:
`instantiation_error`
`Frame` is not a valid CCSDS transfer frame term for the selected frame type:
`domain_error(ccsds_frame_term,Frame)`

---

verify_fecf/1

True if the frame term contains the correct FECF for the selected object configuration.

Compilation flags:
`static`

Template:
`verify_fecf(Frame)`
Mode and number of proofs:
`verify_fecf(@compound)` - `zero_or_one`

---

frame_type/2

Returns the transfer frame type for a frame term.

Compilation flags:
`static`

Template:
`frame_type(Frame,Type)`
Mode and number of proofs:
`frame_type(+compound,-atom)` - `one`

---

version/2

Extracts the transfer frame version number.

Compilation flags:
`static`

Template:
`version(Frame,Version)`
Mode and number of proofs:
`version(+compound,-integer)` - `one`

---

spacecraft_id/2

Extracts the spacecraft identifier.

Compilation flags:
```
static
```

Template:
```
spacecraft_id(Frame,SpacecraftId)
```
Mode and number of proofs:
```
spacecraft_id(+compound,-integer) - one
```

---

virtual_channel_id/2

Extracts the virtual channel identifier.

Compilation flags:
```
static
```

Template:
```
virtual_channel_id(Frame,VirtualChannelId)
```
Mode and number of proofs:
```
virtual_channel_id(+compound,-integer) - one
```

---

data_field/2

Extracts the transfer frame data field as a list of bytes.

Compilation flags:
```
static
```

Template:
```
data_field(Frame,DataField)
```
Mode and number of proofs:
```
data_field(+compound,-list(byte)) - one
```

---

ocf/2

Extracts the operational control field. Returns `none` when absent.

Compilation flags:
`static`

Template:
`ocf(Frame,OCF)`
Mode and number of proofs:
`ocf(+compound,-compound)` - `one`

---

fecf/2

Extracts the frame error control field. Returns `none` when absent.

Compilation flags:
`static`

Template:
`fecf(Frame,FECF)`
Mode and number of proofs:
`fecf(+compound,-compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.31.3 ccsds_frames

Helpers for introspecting CCSDS transfer frame terms.

Availability:
`logtalk_load(ccsds_frames(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-05-09

Compilation flags:
`static, context_switching_calls`

Uses:
ccsds_packets(SecondaryHeaderLength)
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *valid/1*
  - *frame_type/2*
  - *version/2*
  - *spacecraft_id/2*
  - *virtual_channel_id/2*
  - *data_field/2*
  - *ocf/2*
  - *fecf/2*
  - *update_fecf/2*
  - *verify_fecf/1*
  - *extract_packets/3*
  - *insert_packets/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

valid/1

True if the argument is a known CCSDS transfer frame term.

Compilation flags:
`static`

Template:
`valid(Frame)`
Mode and number of proofs:
`valid(@compound)` - `zero_or_one`

frame_type/2

Returns the CCSDS transfer frame type for a known frame term.

Compilation flags:
```
static
```

Template:
```
frame_type(Frame,Type)
```
Mode and number of proofs:
```
frame_type(+compound,-atom) - one
```

---

version/2

Extracts the transfer frame version number from a known frame term.

Compilation flags:
```
static
```

Template:
```
version(Frame,Version)
```
Mode and number of proofs:
```
version(+compound,-integer) - one
```

---

spacecraft_id/2

Extracts the spacecraft identifier from a known frame term.

Compilation flags:
```
static
```

Template:
```
spacecraft_id(Frame,SpacecraftId)
```
Mode and number of proofs:
```
spacecraft_id(+compound,-integer) - one
```

---

virtual_channel_id/2

Extracts the virtual channel identifier from a known frame term.

Compilation flags:
`static`

Template:
`virtual_channel_id(Frame,VirtualChannelId)`
Mode and number of proofs:
`virtual_channel_id(+compound,-integer)` - `one`

---

data_field/2

Extracts the raw data field bytes from a known frame term.

Compilation flags:
`static`

Template:
`data_field(Frame,DataField)`
Mode and number of proofs:
`data_field(+compound,-list(byte))` - `one`

---

ocf/2

Extracts the operational control field. Returns `none` when absent.

Compilation flags:
`static`

Template:
`ocf(Frame,OCF)`
Mode and number of proofs:
`ocf(+compound,-compound)` - `one`

---

fecf/2

Extracts the frame error control field. Returns `none` when absent.

Compilation flags:
`static`

Template:
`fecf(Frame,FECF)`
Mode and number of proofs:
`fecf(+compound,-compound)` - `one`

---

update_fecf/2

Refreshes a present frame error control field from the remaining frame fields while leaving frames without an FECF unchanged.

Compilation flags:
`static`

Template:
`update_fecf(Frame,UpdatedFrame)`
Mode and number of proofs:
`update_fecf(+compound,-compound)` - `one_or_error`

Exceptions:
`Frame` is a variable:
`instantiation_error`
`Frame` is not a valid CCSDS transfer frame term:
`domain_error(ccsds_frame_term,Frame)`

---

verify_fecf/1

True if a present frame error control field matches the remaining frame fields. Frames without an FECF succeed.

Compilation flags:
`static`

Template:
`verify_fecf(Frame)`
Mode and number of proofs:
`verify_fecf(@compound)` - `zero_or_one`

---

extract_packets/3

Parses the raw frame data field as CCSDS space packet bytes using the given packet secondary header length.

Compilation flags:
`static`

Template:
`extract_packets(Frame,SecondaryHeaderLength,Packets)`
Mode and number of proofs:
`extract_packets(+compound,+integer,-list(compound))` - `one_or_error`

Exceptions:
`Frame` is a variable:
`instantiation_error`
`SecondaryHeaderLength` is a variable:
`instantiation_error`
`Frame` is neither a variable nor a valid CCSDS transfer frame term:
`domain_error(ccsds_frame_term,Frame)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`
The frame data field is not a valid CCSDS packet byte sequence:
`domain_error(ccsds_byte_sequence,Bytes)`

---

insert_packets/4

Generates CCSDS packet bytes and replaces the raw data field of a frame term, preserving the remaining header fields unchanged.

Compilation flags:
`static`

Template:
`insert_packets(Packets,SecondaryHeaderLength,Frame,UpdatedFrame)`
Mode and number of proofs:
`insert_packets(+list(compound),+integer,+compound,-compound)` - `one_or_error`

Exceptions:
`Packets` is a variable:
`instantiation_error`
`SecondaryHeaderLength` is a variable:
`instantiation_error`
`Frame` is a variable:
`instantiation_error`
`Frame` is neither a variable nor a valid CCSDS transfer frame term:
`domain_error(ccsds_frame_term,Frame)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

An element `Packet` of `Packets` is a variable or `Packets` is a partial list:
`instantiation_error`
An element `Packet` of `Packets` is neither a variable nor a valid CCSDS packet term:
`domain_error(ccsds_packet_term,Packet)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.31.4 `ccsds_frames_types`

Type definitions for CCSDS transfer frames.

Availability:
`logtalk_load(ccsds_frames(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-08

Compilation flags:
`static`

Provides:
type::type/1
type::check/2
Uses:
ccsds_aos_frames(FrameLength,InsertZoneLength,HasOCF,HasFECF)
ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF)
ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.31.5 `ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF)`

- `FrameLength` - Fixed telecommand transfer frame length in octets.
- `SegmentHeaderLength` - Length in octets of the optional segment header when present.
- `HasFECF` - Boolean indicating if frames include the frame error control field.

CCSDS telecommand transfer frame parser and generator.

Availability:
`logtalk_load(ccsds_frames(loader))`

Author: Paulo Moura
Version: 1:0:2
Date: 2026-06-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ccsds_frame_protocol*
Uses:
list
reader
type

Remarks:
(none)

Inherited public predicates:

data_field/2 *fecf/2* *frame_type/2* *generate/2* *generate/3* *ocf/2* *parse/2* *spacecraft_id/2* *valid/1* *version/2* *virtual_channel_id/2*

- *Public predicates*
  - *bypass_flag/2*
  - *control_command_flag/2*
  - *sequence_number/2*
  - *segment_header/2*
  - *update_fecf/2*
  - *verify_fecf/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

bypass_flag/2

Extracts the bypass flag value.

Compilation flags:

```
static
```

Template:

```
bypass_flag(Frame,Flag)
```

Mode and number of proofs:

```
bypass_flag(+compound,-integer) - one
```

---

control_command_flag/2

Extracts the control command flag value.

Compilation flags:

```
static
```

Template:

```
control_command_flag(Frame,Flag)
```

Mode and number of proofs:

```
control_command_flag(+compound,-integer) - one
```

---

sequence_number/2

Extracts the frame sequence number.

Compilation flags:
`static`

Template:
`sequence_number(Frame,SequenceNumber)`
Mode and number of proofs:
`sequence_number(+compound,-integer)` - `one`

---

segment_header/2

Extracts the optional segment header. Returns `none` when absent.

Compilation flags:
`static`

Template:
`segment_header(Frame,SegmentHeader)`
Mode and number of proofs:
`segment_header(+compound,-compound)` - `one`

---

update_fecf/2

Computes the correct telecommand transfer frame FECF for the selected object configuration and returns the corresponding updated frame term.

Compilation flags:
`static`

Template:
`update_fecf(Frame,UpdatedFrame)`
Mode and number of proofs:
`update_fecf(+compound,-compound)` - `one_or_error`

Exceptions:
`Frame` is a variable:
`instantiation_error`
`Frame` is neither a variable nor a valid telecommand transfer frame term:
`domain_error(ccsds_frame_term,Frame)`

`verify_fecf/1`

True if the telecommand transfer frame term contains the correct FECF for the selected object configuration.

Compilation flags:
`static`

Template:
`verify_fecf(Frame)`
Mode and number of proofs:
`verify_fecf(@compound)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.31.6 `ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)`

- `FrameLength` - Fixed telemetry transfer frame length in octets.
- `SecondaryHeaderLength` - Length in octets of the telemetry transfer frame secondary header when present.
- `HasFECF` - Boolean indicating if frames include the frame error control field.

CCSDS telemetry transfer frame parser and generator.

Availability:
`logtalk_load(ccsds_frames(loader))`

Author: Paulo Moura
Version: 1:0:2
Date: 2026-06-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ccsds_frame_protocol*
Uses:
list
reader
type

Remarks:
(none)

Inherited public predicates:
data_field/2 *fecf/2* *frame_type/2* *generate/2* *generate/3* *ocf/2* *parse/2* *spacecraft_id/2* *valid/1* *version/2* *virtual_channel_id/2*

- *Public predicates*
  - *ocf_flag/2*
  - *master_channel_frame_count/2*
  - *virtual_channel_frame_count/2*
  - *secondary_header_flag/2*
  - *synchronization_flag/2*
  - *packet_order_flag/2*
  - *segment_length_identifier/2*
  - *first_header_pointer/2*
  - *secondary_header/2*
  - *update_fecf/2*
  - *verify_fecf/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

ocf_flag/2

Extracts the operational control field flag. Returns `absent` or `present`.

Compilation flags:
`static`

Template:
`ocf_flag(Frame,Flag)`

Mode and number of proofs:
`ocf_flag(+compound,-atom)` - `one`

master_channel_frame_count/2

Extracts the master channel frame count.

Compilation flags:
`static`

Template:
`master_channel_frame_count(Frame,Count)`
Mode and number of proofs:
`master_channel_frame_count(+compound,-integer)` - `one`

---

virtual_channel_frame_count/2

Extracts the virtual channel frame count.

Compilation flags:
`static`

Template:
`virtual_channel_frame_count(Frame,Count)`
Mode and number of proofs:
`virtual_channel_frame_count(+compound,-integer)` - `one`

---

secondary_header_flag/2

Extracts the telemetry transfer frame secondary header flag. Returns `absent` or `present`.

Compilation flags:
`static`

Template:
`secondary_header_flag(Frame,Flag)`
Mode and number of proofs:
`secondary_header_flag(+compound,-atom)` - `one`

---

synchronization_flag/2

Extracts the synchronization flag value.

Compilation flags:

```
static
```

Template:

```
synchronization_flag(Frame,Flag)
```

Mode and number of proofs:

```
synchronization_flag(+compound,-integer) - one
```

---

packet_order_flag/2

Extracts the packet order flag value.

Compilation flags:

```
static
```

Template:

```
packet_order_flag(Frame,Flag)
```

Mode and number of proofs:

```
packet_order_flag(+compound,-integer) - one
```

---

segment_length_identifier/2

Extracts the segment length identifier value.

Compilation flags:

```
static
```

Template:

```
segment_length_identifier(Frame,Identifier)
```

Mode and number of proofs:

```
segment_length_identifier(+compound,-integer) - one
```

---

first_header_pointer/2

Extracts the first header pointer value.

Compilation flags:
`static`

Template:
`first_header_pointer(Frame,Pointer)`
Mode and number of proofs:
`first_header_pointer(+compound,-integer)` - `one`

---

secondary_header/2

Extracts the telemetry transfer frame secondary header. Returns `none` when absent.

Compilation flags:
`static`

Template:
`secondary_header(Frame,SecondaryHeader)`
Mode and number of proofs:
`secondary_header(+compound,-compound)` - `one`

---

update_fecf/2

Computes the correct telemetry transfer frame FECF for the selected object configuration and returns the corresponding updated frame term.

Compilation flags:
`static`

Template:
`update_fecf(Frame,UpdatedFrame)`
Mode and number of proofs:
`update_fecf(+compound,-compound)` - `one_or_error`

Exceptions:
`Frame` is a variable:
`instantiation_error`
`Frame` is neither a variable nor a valid telemetry transfer frame term:
`domain_error(ccsds_frame_term,Frame)`

---

verify_fecf/1

True if the telemetry transfer frame term contains the correct FECF for the selected object configuration.

Compilation flags:
`static`

Template:
`verify_fecf(Frame)`
Mode and number of proofs:
`verify_fecf(@compound)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.32 ccsds_link_profiles

object

1.32.1 ccsds_link_profiles

Ergonomic wrappers around the CCSDS TM, TC, and AOS frame objects using explicit profile terms.

Availability:
`logtalk_load(ccsds_link_profiles(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-20

Compilation flags:
`static, context_switching_calls`

Uses:
ccsds_packet_services

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *valid_profile/1*
  - *parse_frame/3*
  - *parse_frames/3*
  - *generate_frame/3*
  - *generate_frames/3*
  - *valid_reassembly_state/1*
  - *initial_reassembly_state/1*
  - *pending_fragments/2*
  - *valid_discontinuity_policy/1*
  - *extract_packets/4*
  - *insert_packets/5*
  - *reassemble_packets/6*
  - *reassemble_packets/7*
  - *reassemble_packets/8*
  - *reassemble_frames/6*
  - *reassemble_frames/7*
  - *reassemble_frames/8*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

valid_profile/1

True if the argument is a supported CCSDS link profile term.

Compilation flags:
`static`

Template:
`valid_profile(Profile)`
Mode and number of proofs:
`valid_profile(@compound)` - `zero_or_one`

parse_frame/3

Parses exactly one CCSDS transfer frame from a source using a link profile term. Supported source terms are `file(File)`, `stream(Stream)`, and `bytes(Bytes)`.

Compilation flags:
`static`

Template:
`parse_frame(Source,Profile,Frame)`
Mode and number of proofs:
`parse_frame(+compound,+compound,-compound)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Profile` is a variable:
`instantiation_error`
`Profile` is neither a variable nor a supported link profile term:
`domain_error(ccsds_link_profile,Profile)`
`Source` does not contain exactly one frame for the selected profile:
`domain_error(ccsds_single_frame_source,Source)`

---

parse_frames/3

Parses zero or more CCSDS transfer frames from a source using a link profile term. Supported source terms are `file(File)`, `stream(Stream)`, and `bytes(Bytes)`.

Compilation flags:
`static`

Template:
`parse_frames(Source,Profile,Frames)`
Mode and number of proofs:
`parse_frames(+compound,+compound,-list(compound))` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Profile` is a variable:
`instantiation_error`
`Profile` is neither a variable nor a supported link profile term:
`domain_error(ccsds_link_profile,Profile)`

---

generate_frame/3

Generates exactly one CCSDS transfer frame to a sink using a link profile term. Supported sink terms are `file(File)`, `stream(Stream)`, and `bytes(Bytes)`.

Compilation flags:
`static`

Template:
`generate_frame(Sink,Profile,Frame)`
Mode and number of proofs:
`generate_frame(+compound,+compound,+compound)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Profile` is a variable:
`instantiation_error`
`Frame` is a variable:
`instantiation_error`
`Profile` is neither a variable nor a supported link profile term:
`domain_error(ccsds_link_profile,Profile)`
`Frame` is neither a variable nor a valid frame term for the selected profile:
`domain_error(ccsds_frame_term,Frame)`

---

generate_frames/3

Generates zero or more CCSDS transfer frames to a sink using a link profile term. Supported sink terms are `file(File)`, `stream(Stream)`, and `bytes(Bytes)`.

Compilation flags:
`static`

Template:
`generate_frames(Sink,Profile,Frames)`
Mode and number of proofs:
`generate_frames(+compound,+compound,+list(compound))` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Profile` is a variable:
`instantiation_error`
`Frames` is a variable:
`instantiation_error`
`Profile` is neither a variable nor a supported link profile term:
`domain_error(ccsds_link_profile,Profile)`
`Frames` is neither a variable nor a list of valid frame terms for the selected profile:

```
domain_error(ccsds_frame_terms,Frames)
```

---

valid_reassembly_state/1

True if the argument is a valid TM/AOS channel packet reassembly state term for the profile-level packet reassembly predicates.

Compilation flags:
`static`

Template:
`valid_reassembly_state(State)`
Mode and number of proofs:
`valid_reassembly_state(@compound)` - `zero_or_one`

---

initial_reassembly_state/1

Returns the initial TM/AOS channel packet reassembly state for the profile-level packet reassembly predicates.

Compilation flags:
`static`

Template:
`initial_reassembly_state(State)`
Mode and number of proofs:
`initial_reassembly_state(-compound)` - `one`

---

pending_fragments/2

Returns the non-empty pending packet fragments currently buffered per TM or AOS virtual channel.

Compilation flags:
`static`

Template:
`pending_fragments(State,PendingFragments)`
Mode and number of proofs:
`pending_fragments(+compound,-list(compound))` - `one_or_error`

Exceptions:
`State` is a variable:
`instantiation_error`
`State` is not a valid channel reassembly state term:
`domain_error(ccsds_channel_reassembly_state,State)`

---

valid_discontinuity_policy/1

True if the argument is a valid packet reassembly discontinuity recovery policy atom. Valid values are `throw`, `drop`, and `resynchronize`.

Compilation flags:
`static`

Template:
`valid_discontinuity_policy(Policy)`
Mode and number of proofs:
`valid_discontinuity_policy(@atom)` - `zero_or_one`

---

extract_packets/4

Extracts the TM first-header-pointer packet zone or the AOS M_PDU packet zone from a TM or AOS frame using the selected link profile and packet secondary header length.

Compilation flags:
`static`

Template:
`extract_packets(Profile,SecondaryHeaderLength,Frame,PacketZone)`
Mode and number of proofs:
`extract_packets(+compound,+integer,+compound,-compound)` - `one_or_error`

Exceptions:
`Profile` is a variable:
`instantiation_error`
`SecondaryHeaderLength` is a variable:
`instantiation_error`
`Frame` is a variable:
`instantiation_error`
`Profile` is neither a variable nor a supported link profile term:
`domain_error(ccsds_link_profile,Profile)`
`Profile` is a telecommand profile and therefore does not support packet extraction:
`domain_error(ccsds_packet_link_profile,Profile)`
`Frame` is neither a variable nor a valid frame term for the selected profile:
`domain_error(ccsds_frame_term,Frame)`

---

insert_packets/5

Regenerates the TM first-header-pointer packet zone or the AOS M_PDU packet zone for a TM or AOS frame using the selected link profile and packet secondary header length.

Compilation flags:

`static`

Template:

`insert_packets(Profile,SecondaryHeaderLength,PacketZone,Frame,UpdatedFrame)`

Mode and number of proofs:

`insert_packets(+compound,+integer,+compound,+compound,-compound)` - `one_or_error`

Exceptions:

`Profile` is a variable:

`instantiation_error`

`SecondaryHeaderLength` is a variable:

`instantiation_error`

`PacketZone` is a variable:

`instantiation_error`

`Frame` is a variable:

`instantiation_error`

`Profile` is neither a variable nor a supported link profile term:

`domain_error(ccsds_link_profile,Profile)`

`Profile` is a telecommand profile and therefore does not support packet insertion:

`domain_error(ccsds_packet_link_profile,Profile)`

`Frame` is neither a variable nor a valid frame term for the selected profile:

`domain_error(ccsds_frame_term,Frame)`

---

reassemble_packets/6

Reassembles complete packets from a single TM or AOS transfer frame using the default `throw` discontinuity recovery policy.

Compilation flags:

`static`

Template:

`reassemble_packets(Profile,SecondaryHeaderLength,Frame,State,Packets,UpdatedState)`

Mode and number of proofs:

`reassemble_packets(+compound,+integer,+compound,+compound,-list(compound),-compound)` - `one_or_error`

Exceptions:

`Profile`, `SecondaryHeaderLength`, `Frame`, or `State` is a variable:

`instantiation_error`

`Profile` is neither a variable nor a supported link profile term:

`domain_error(ccsds_link_profile,Profile)`

`Profile` is a telecommand profile and therefore does not support packet reassembly:
`domain_error(ccsds_packet_link_profile,Profile)`
`Frame` is neither a variable nor a valid TM or AOS frame term for the selected profile:
`domain_error(ccsds_frame_term,Frame)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`
`State` is neither a variable nor a valid channel reassembly state term:
`domain_error(ccsds_channel_reassembly_state,State)`
The delegated packet reassembly detects a discontinuous frame sequence, inconsistent counter modulus, or invalid continuation state:
`domain_error(ccsds_packet_reassembly_continuation, packet_reassembly_state(PendingData))`
A delegated packet continuation fragment is followed by a new packet before the previous packet is completed:
`domain_error(ccsds_packet_reassembly_continuation, incomplete_continuation_before_new_packet)`

`reassemble_packets/7`

Reassembles complete packets from a single TM or AOS transfer frame using the selected discontinuity recovery policy.

Compilation flags:
`static`

Template:
`reassemble_packets(Profile,SecondaryHeaderLength,Frame,Policy,State,Packets, UpdatedState)`
Mode and number of proofs:
`reassemble_packets(+compound,+integer,+compound,+atom,+compound,-list(compound), -compound)` - `one_or_error`

Exceptions:
`Profile`, `SecondaryHeaderLength`, `Frame`, `Policy`, or `State` is a variable:
`instantiation_error`
`Profile` is neither a variable nor a supported link profile term:
`domain_error(ccsds_link_profile,Profile)`
`Profile` is a telecommand profile and therefore does not support packet reassembly:
`domain_error(ccsds_packet_link_profile,Profile)`
`Frame` is neither a variable nor a valid TM or AOS frame term for the selected profile:
`domain_error(ccsds_frame_term,Frame)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`
`Policy` is neither a variable nor a valid discontinuity recovery policy:
`domain_error(ccsds_discontinuity_policy,Policy)`
`State` is neither a variable nor a valid channel reassembly state term:
`domain_error(ccsds_channel_reassembly_state,State)`
The delegated packet reassembly detects a discontinuous frame sequence, inconsistent counter modulus, or invalid continuation state under the `throw` policy:

`domain_error(ccsds_packet_reassembly_continuation, packet_reassembly_state(PendingData))`

A delegated packet continuation fragment is followed by a new packet before the previous packet is completed:

`domain_error(ccsds_packet_reassembly_continuation, incomplete_continuation_before_new_packet)`

`reassemble_packets/8`

Reassembles complete packets from a single TM or AOS transfer frame using the selected discontinuity recovery policy and returns any recovery events.

Compilation flags:

`static`

Template:

`reassemble_packets(Profile,SecondaryHeaderLength,Frame,Policy,State,Packets,UpdatedState,Events)`

Mode and number of proofs:

`reassemble_packets(+compound,+integer,+compound,+atom,+compound,-list(compound),-compound,-list(compound))` - `one_or_error`

Exceptions:

`Profile`, `SecondaryHeaderLength`, `Frame`, `Policy`, or `State` is a variable:

`instantiation_error`

`Profile` is neither a variable nor a supported link profile term:

`domain_error(ccsds_link_profile,Profile)`

`Profile` is a telecommand profile and therefore does not support packet reassembly:

`domain_error(ccsds_packet_link_profile,Profile)`

`Frame` is neither a variable nor a valid TM or AOS frame term for the selected profile:

`domain_error(ccsds_frame_term,Frame)`

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

`Policy` is neither a variable nor a valid discontinuity recovery policy:

`domain_error(ccsds_discontinuity_policy,Policy)`

`State` is neither a variable nor a valid channel reassembly state term:

`domain_error(ccsds_channel_reassembly_state,State)`

The delegated packet reassembly detects a discontinuous frame sequence, inconsistent counter modulus, or invalid continuation state under the `throw` policy:

`domain_error(ccsds_packet_reassembly_continuation, packet_reassembly_state(PendingData))`

A delegated packet continuation fragment is followed by a new packet before the previous packet is completed:

`domain_error(ccsds_packet_reassembly_continuation, incomplete_continuation_before_new_packet)`

---

reassemble_frames/6

Reassembles complete packets across a sequence of TM or AOS transfer frames using the default `throw` discontinuity recovery policy.

Compilation flags:

`static`

Template:

```
reassemble_frames(Profile,SecondaryHeaderLength,Frames,State,Packets,
UpdatedState)
```

Mode and number of proofs:

```
reassemble_frames(+compound,+integer,+list(compound),+compound,-list(compound),
-compound) - one_or_error
```

Exceptions:

`Profile`, `SecondaryHeaderLength`, `Frames`, or `State` is a variable:

```
instantiation_error
```

`Profile` is neither a variable nor a supported link profile term:

```
domain_error(ccsds_link_profile,Profile)
```

`Profile` is a telecommand profile and therefore does not support packet reassembly:

```
domain_error(ccsds_packet_link_profile,Profile)
```

`Frames` is neither a variable nor a list of valid TM or AOS frame terms for the selected profile:

```
domain_error(ccsds_frame_terms,Frames)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`State` is neither a variable nor a valid channel reassembly state term:

```
domain_error(ccsds_channel_reassembly_state,State)
```

The delegated packet reassembly detects a discontinuous frame sequence, inconsistent counter modulus, or invalid continuation state:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A delegated packet continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

reassemble_frames/7

Reassembles complete packets across a sequence of TM or AOS transfer frames using the selected discontinuity recovery policy.

Compilation flags:

`static`

Template:

```
reassemble_frames(Profile,SecondaryHeaderLength,Frames,Policy,State,Packets,
UpdatedState)
```

Mode and number of proofs:

```
reassemble_frames(+compound,+integer,+list(compound),+atom,+compound,
-list(compound),-compound) - one_or_error
```

Exceptions:

`Profile`, `SecondaryHeaderLength`, `Frames`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`Profile` is neither a variable nor a supported link profile term:

```
domain_error(ccsds_link_profile,Profile)
```

`Profile` is a telecommand profile and therefore does not support packet reassembly:

```
domain_error(ccsds_packet_link_profile,Profile)
```

`Frames` is neither a variable nor a list of TM or AOS frame terms for the selected profile:

```
domain_error(ccsds_frame_terms,Frames)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`Policy` is neither a variable nor a valid discontinuity recovery policy:

```
domain_error(ccsds_discontinuity_policy,Policy)
```

`State` is neither a variable nor a valid channel reassembly state term:

```
domain_error(ccsds_channel_reassembly_state,State)
```

The delegated packet reassembly detects a discontinuous frame sequence, inconsistent counter modulus, or invalid continuation state under the `throw` policy:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A delegated packet continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

`reassemble_frames/8`

Reassembles complete packets across a sequence of TM or AOS transfer frames using the selected discontinuity recovery policy and returns any recovery events.

Compilation flags:

```
static
```

Template:

```
reassemble_frames(Profile,SecondaryHeaderLength,Frames,Policy,State,Packets,
UpdatedState,Events)
```

Mode and number of proofs:

```
reassemble_frames(+compound,+integer,+list(compound),+atom,+compound,
-list(compound),-compound,-list(compound)) - one_or_error
```

Exceptions:

`Profile`, `SecondaryHeaderLength`, `Frames`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`Profile` is neither a variable nor a supported link profile term:

```
domain_error(ccsds_link_profile,Profile)
```

`Profile` is a telecommand profile and therefore does not support packet reassembly:

```
domain_error(ccsds_packet_link_profile,Profile)
```

`Frames` is neither a variable nor a list of TM or AOS frame terms for the selected profile:

```
domain_error(ccsds_frame_terms,Frames)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

`Policy` is neither a variable nor a valid discontinuity recovery policy:

`domain_error(ccsds_discontinuity_policy,Policy)`

`State` is neither a variable nor a valid channel reassembly state term:

`domain_error(ccsds_channel_reassembly_state,State)`

The delegated packet reassembly detects a discontinuous frame sequence, inconsistent counter modulus, or invalid continuation state under the `throw` policy:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A delegated packet continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.33 ccsds_packet_services

object

1.33.1 `ccsds_packet_services`

Helpers for CCSDS packet service packet-zone splitting and cross-frame TM/AOS packet reassembly.

Availability:

`logtalk_load(ccsds_packet_services(loader))`

Author: Paulo Moura
Version: 0:1:1
Date: 2026-06-14

Compilation flags:

`static, context_switching_calls`

Uses:

ccsds_packets(SecondaryHeaderLength)
list
type

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *valid_reassembly_state/1*
  - *initial_reassembly_state/1*
  - *pending_data/2*
  - *valid_channel_reassembly_state/1*
  - *initial_channel_reassembly_state/1*
  - *pending_fragments/2*
  - *valid_discontinuity_policy/1*
  - *valid/1*
  - *prefix_data/2*
  - *packets/2*
  - *suffix_data/2*
  - *split_packet_zone/4*
  - *join_packet_zone/4*
  - *extract_tm_packets/3*
  - *insert_tm_packets/4*
  - *split_aos_packet_zone/3*
  - *join_aos_packet_zone/3*
  - *reassemble_packet_zone/5*
  - *extract_aos_packets/3*
  - *insert_aos_packets/4*
  - *reassemble_tm_packets/5*
  - *reassemble_tm_packets/6*
  - *reassemble_tm_packets/7*
  - *reassemble_tm_frames/5*
  - *reassemble_tm_frames/6*
  - *reassemble_tm_frames/7*
  - *reassemble_aos_packets/5*
  - *reassemble_aos_packets/6*
  - *reassemble_aos_packets/7*
  - *reassemble_aos_frames/5*
  - *reassemble_aos_frames/6*
  - *reassemble_aos_frames/7*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

valid_reassembly_state/1

True if the argument is a valid packet reassembly state term.

Compilation flags:
```
static
```

Template:
```
valid_reassembly_state(State)
```
Mode and number of proofs:
```
valid_reassembly_state(@compound) - zero_or_one
```

---

initial_reassembly_state/1

Returns the initial cross-frame packet reassembly state.

Compilation flags:
```
static
```

Template:
```
initial_reassembly_state(State)
```
Mode and number of proofs:
```
initial_reassembly_state(-compound) - one
```

---

pending_data/2

Extracts the currently buffered trailing packet fragment bytes from a packet reassembly state.

Compilation flags:
```
static
```

Template:
```
pending_data(State,Bytes)
```
Mode and number of proofs:
```
pending_data(+compound,-list(byte)) - one
```

---

valid_channel_reassembly_state/1

True if the argument is a valid frame-channel packet reassembly state term keyed by frame type, spacecraft identifier, and virtual channel identifier.

Compilation flags:

```
static
```

Template:

```
valid_channel_reassembly_state(State)
```

Mode and number of proofs:

```
valid_channel_reassembly_state(@compound) - zero_or_one
```

---

initial_channel_reassembly_state/1

Returns the initial frame-channel packet reassembly state.

Compilation flags:

```
static
```

Template:

```
initial_channel_reassembly_state(State)
```

Mode and number of proofs:

```
initial_channel_reassembly_state(-compound) - one
```

---

pending_fragments/2

Extracts the non-empty pending packet fragments buffered per frame type, spacecraft identifier, and virtual channel identifier.

Compilation flags:

```
static
```

Template:

```
pending_fragments(State,PendingFragments)
```

Mode and number of proofs:

```
pending_fragments(+compound,-list(compound)) - one
```

---

valid_discontinuity_policy/1

True if the argument is a valid discontinuity recovery policy atom. Valid values are `throw`, `drop`, and `resynchronize`.

Compilation flags:
`static`

Template:
`valid_discontinuity_policy(Policy)`
Mode and number of proofs:
`valid_discontinuity_policy(@atom)` - `zero_or_one`

---

valid/1

True if the argument is a valid packet-zone term.

Compilation flags:
`static`

Template:
`valid(PacketZone)`
Mode and number of proofs:
`valid(@compound)` - `zero_or_one`

---

prefix_data/2

Extracts the bytes that precede the first complete packet in a packet zone.

Compilation flags:
`static`

Template:
`prefix_data(PacketZone,PrefixData)`
Mode and number of proofs:
`prefix_data(+compound,-list(byte))` - `one`

---

packets/2

Extracts the list of complete packets in a packet zone.

Compilation flags:
`static`

Template:
`packets(PacketZone,Packets)`
Mode and number of proofs:
`packets(+compound,-list(compound))` - `one`

---

suffix_data/2

Extracts the bytes that trail the last complete packet in a packet zone.

Compilation flags:
`static`

Template:
`suffix_data(PacketZone,SuffixData)`
Mode and number of proofs:
`suffix_data(+compound,-list(byte))` - `one`

---

split_packet_zone/4

Splits a packet-zone byte sequence using the first header pointer and parses any complete packets starting at that offset. A pointer value of 2047 denotes that no packet starts in the zone.

Compilation flags:
`static`

Template:
`split_packet_zone(Bytes,FirstHeaderPointer,SecondaryHeaderLength,PacketZone)`
Mode and number of proofs:
`split_packet_zone(+list(byte),+integer,+integer,-compound)` - `one_or_error`

Exceptions:
`Bytes`, `FirstHeaderPointer`, or `SecondaryHeaderLength` is a variable:
`instantiation_error`
`Bytes` is neither a variable nor a valid packet-zone byte sequence:
`domain_error(ccsds_packet_zone_bytes,Bytes)`
`FirstHeaderPointer` is neither a variable nor a valid first header pointer for `Bytes`:
`domain_error(ccsds_first_header_pointer,FirstHeaderPointer)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

---

join_packet_zone/4

Encodes a packet-zone term back into bytes and returns the corresponding first header pointer. When the packet list is empty, the pointer is set to 2047.

Compilation flags:
`static`

Template:
`join_packet_zone(PacketZone,SecondaryHeaderLength,Bytes,FirstHeaderPointer)`
Mode and number of proofs:
`join_packet_zone(+compound,+integer,-list(byte),-integer)` - `one_or_error`

Exceptions:
`PacketZone` or `SecondaryHeaderLength` is a variable:
`instantiation_error`
`PacketZone` is neither a variable nor a valid packet-zone term:
`domain_error(ccsds_packet_zone,PacketZone)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

---

extract_tm_packets/3

Extracts a TM transfer frame packet zone as a packet-zone term using the frame first header pointer and the given packet secondary header length.

Compilation flags:
`static`

Template:
`extract_tm_packets(Frame,SecondaryHeaderLength,PacketZone)`
Mode and number of proofs:
`extract_tm_packets(+compound,+integer,-compound)` - `one_or_error`

Exceptions:
`Frame` or `SecondaryHeaderLength` is a variable:
`instantiation_error`
`Frame` is neither a variable nor a valid TM transfer frame term:
`domain_error(ccsds_tm_transfer_frame_term,Frame)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`
The frame packet-zone bytes or first header pointer are invalid:
`domain_error(ccsds_packet_zone_bytes,Bytes)`

---

insert_tm_packets/4

Updates a TM transfer frame data field and first header pointer from a packet-zone term while preserving the remaining frame fields unchanged.

Compilation flags:

```
static
```

Template:

```
insert_tm_packets(PacketZone,SecondaryHeaderLength,Frame,UpdatedFrame)
```

Mode and number of proofs:

```
insert_tm_packets(+compound,+integer,+compound,-compound) - one_or_error
```

Exceptions:

`PacketZone`, `SecondaryHeaderLength`, or `Frame` is a variable:

```
instantiation_error
```

`PacketZone` is neither a variable nor a valid packet-zone term:

```
domain_error(ccsds_packet_zone,PacketZone)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`Frame` is neither a variable nor a valid TM transfer frame term:

```
domain_error(ccsds_tm_transfer_frame_term,Frame)
```

---

split_aos_packet_zone/3

Splits an AOS M_PDU packet-service data field into a packet-zone term. The first two octets are interpreted as the M_PDU first header pointer field, where 2046 denotes idle data only and 2047 denotes that no packet starts in the packet zone.

Compilation flags:

```
static
```

Template:

```
split_aos_packet_zone(Bytes,SecondaryHeaderLength,PacketZone)
```

Mode and number of proofs:

```
split_aos_packet_zone(+list(byte),+integer,-compound) - one_or_error
```

Exceptions:

`Bytes` or `SecondaryHeaderLength` is a variable:

```
instantiation_error
```

`Bytes` is neither a variable nor a valid AOS packet-zone byte sequence:

```
domain_error(ccsds_aos_packet_zone_bytes,Bytes)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

---

join_aos_packet_zone/3

Encodes a packet-zone term as an AOS M_PDU packet-service data field, including the two-octet first header pointer field.

Compilation flags:
`static`

Template:
`join_aos_packet_zone(PacketZone,SecondaryHeaderLength,Bytes)`
Mode and number of proofs:
`join_aos_packet_zone(+compound,+integer,-list(byte))` - `one_or_error`

Exceptions:
`PacketZone` or `SecondaryHeaderLength` is a variable:
`instantiation_error`
`PacketZone` is neither a variable nor a valid packet-zone term:
`domain_error(ccsds_packet_zone,PacketZone)`
`PacketZone` has neither packets nor a pure prefix or suffix and therefore cannot be encoded as an empty AOS packet zone:
`domain_error(ccsds_aos_packet_zone,PacketZone)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

---

reassemble_packet_zone/5

Reassembles complete packets from a packet-zone term and a prior packet reassembly state, returning any complete packets plus the updated trailing fragment state.

Compilation flags:
`static`

Template:
`reassemble_packet_zone(PacketZone,SecondaryHeaderLength,State,Packets,UpdatedState)`
Mode and number of proofs:
`reassemble_packet_zone(+compound,+integer,+compound,-list(compound),-compound)` - `one_or_error`

Exceptions:
`PacketZone`, `SecondaryHeaderLength`, or `State` is a variable:
`instantiation_error`
`PacketZone` is neither a variable nor a valid packet-zone term:
`domain_error(ccsds_packet_zone,PacketZone)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`
`State` is neither a variable nor a valid packet reassembly state:
`domain_error(ccsds_packet_reassembly_state,State)`
A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

extract_aos_packets/3

Extracts an AOS transfer frame packet zone as a packet-zone term using the two-octet M_PDU header at the beginning of the frame data field.

Compilation flags:

```
static
```

Template:

```
extract_aos_packets(Frame,SecondaryHeaderLength,PacketZone)
```

Mode and number of proofs:

```
extract_aos_packets(+compound,+integer,-compound) - one_or_error
```

Exceptions:

`Frame` or `SecondaryHeaderLength` is a variable:

```
instantiation_error
```

`Frame` is neither a variable nor a valid AOS transfer frame term:

```
domain_error(ccsds_aos_transfer_frame_term,Frame)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

The frame data field is not a valid AOS packet-zone byte sequence:

```
domain_error(ccsds_aos_packet_zone_bytes,Bytes)
```

---

insert_aos_packets/4

Updates an AOS transfer frame data field from a packet-zone term by regenerating the M_PDU first header pointer and packet-zone bytes while preserving the remaining frame fields unchanged.

Compilation flags:

```
static
```

Template:

```
insert_aos_packets(PacketZone,SecondaryHeaderLength,Frame,UpdatedFrame)
```

Mode and number of proofs:

```
insert_aos_packets(+compound,+integer,+compound,-compound) - one_or_error
```

Exceptions:

`PacketZone`, `SecondaryHeaderLength`, or `Frame` is a variable:

```
instantiation_error
```

`PacketZone` is neither a variable nor a valid packet-zone term:

```
domain_error(ccsds_packet_zone,PacketZone)
```

`PacketZone` cannot be encoded as an AOS empty packet zone:

```
domain_error(ccsds_aos_packet_zone,PacketZone)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`Frame` is neither a variable nor a valid AOS transfer frame term:

```
domain_error(ccsds_aos_transfer_frame_term,Frame)
```

---

reassemble_tm_packets/5

Extracts the packet zone from a TM transfer frame, using the default `throw` discontinuity recovery policy, and returns the complete packets plus the updated channel reassembly state.

Compilation flags:

```
static
```

Template:

```
reassemble_tm_packets(Frame,SecondaryHeaderLength,State,Packets,UpdatedState)
```

Mode and number of proofs:

```
reassemble_tm_packets(+compound,+integer,+compound,-list(compound),-compound) -
one_or_error
```

Exceptions:

`Frame`, `SecondaryHeaderLength`, or `State` is a variable:

```
instantiation_error
```

`Frame` is neither a variable nor a valid TM transfer frame term:

```
domain_error(ccsds_tm_transfer_frame_term,Frame)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`State` is neither a variable nor a valid channel reassembly state term:

```
domain_error(ccsds_channel_reassembly_state,State)
```

The TM frame sequence is discontinuous:

```
domain_error(ccsds_tm_transfer_frame_sequence,
tm_transfer_frame_sequence(SpacecraftId,VirtualChannelId,ExpectedFrameCount,
VirtualChannelFrameCount))
```

The stored and current TM frame counter moduli differ:

```
domain_error(ccsds_tm_transfer_frame_counter_modulus,
tm_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

reassemble_tm_packets/6

Extracts the packet zone from a TM transfer frame, applies the selected discontinuity recovery policy when frame-count continuity is broken, and returns the complete packets plus the updated channel reassembly state.

Compilation flags:

```
static
```

Template:

```
reassemble_tm_packets(Frame,SecondaryHeaderLength,Policy,State,Packets,
UpdatedState)
```

Mode and number of proofs:

```
reassemble_tm_packets(+compound,+integer,+atom,+compound,-list(compound),
-compound) - one_or_error
```

Exceptions:

`Frame`, `SecondaryHeaderLength`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`Frame` is neither a variable nor a valid TM transfer frame term:

```
domain_error(ccsds_tm_transfer_frame_term,Frame)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`Policy` is neither a variable nor a valid discontinuity recovery policy:

```
domain_error(ccsds_discontinuity_policy,Policy)
```

`State` is neither a variable nor a valid channel reassembly state term:

```
domain_error(ccsds_channel_reassembly_state,State)
```

The TM frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_tm_transfer_frame_sequence,
tm_transfer_frame_sequence(SpacecraftId,VirtualChannelId,ExpectedFrameCount,
VirtualChannelFrameCount))
```

The stored and current TM frame counter moduli differ:

```
domain_error(ccsds_tm_transfer_frame_counter_modulus,
tm_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

reassemble_tm_packets/7

Extracts the packet zone from a TM transfer frame, applies the selected discontinuity recovery policy when frame-count continuity is broken, and returns the complete packets, updated channel reassembly state, and explicit recovery events.

Compilation flags:

```
static
```

Template:

```
reassemble_tm_packets(Frame,SecondaryHeaderLength,Policy,State,Packets,
UpdatedState,Events)
```

Mode and number of proofs:

```
reassemble_tm_packets(+compound,+integer,+atom,+compound,-list(compound),
-compound,-list(compound)) - one_or_error
```

Exceptions:

`Frame`, `SecondaryHeaderLength`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`Frame` is neither a variable nor a valid TM transfer frame term:

```
domain_error(ccsds_tm_transfer_frame_term,Frame)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`Policy` is neither a variable nor a valid discontinuity recovery policy:

```
domain_error(ccsds_discontinuity_policy,Policy)
```

`State` is neither a variable nor a valid channel reassembly state term:

```
domain_error(ccsds_channel_reassembly_state,State)
```

The TM frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_tm_transfer_frame_sequence,
tm_transfer_frame_sequence(SpacecraftId,VirtualChannelId,ExpectedFrameCount,
VirtualChannelFrameCount))
```

The stored and current TM frame counter moduli differ:

```
domain_error(ccsds_tm_transfer_frame_counter_modulus,
tm_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

reassemble_tm_frames/5

Reassembles complete packets across a sequence of TM transfer frames using the default `throw` discontinuity recovery policy.

Compilation flags:

```
static
```

Template:

```
reassemble_tm_frames(Frames,SecondaryHeaderLength,State,Packets,UpdatedState)
```

Mode and number of proofs:

```
reassemble_tm_frames(+list(compound),+integer,+compound,-list(compound),
-compound) - one_or_error
```

Exceptions:

`Frames`, `SecondaryHeaderLength`, or `State` is a variable:

```
instantiation_error
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`State` is neither a variable nor a valid channel reassembly state term:

```
domain_error(ccsds_channel_reassembly_state,State)
```

An element `Frame` of `Frames` is neither a variable nor a valid TM transfer frame term:

```
domain_error(ccsds_tm_transfer_frame_term,Frame)
```

The TM frame sequence is discontinuous:

```
domain_error(ccsds_tm_transfer_frame_sequence,
tm_transfer_frame_sequence(SpacecraftId,VirtualChannelId,ExpectedFrameCount,
VirtualChannelFrameCount))
```

The stored and current TM frame counter moduli differ:

```
domain_error(ccsds_tm_transfer_frame_counter_modulus,
tm_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

reassemble_tm_frames/6

Reassembles complete packets across a sequence of TM transfer frames using the selected discontinuity recovery policy.

Compilation flags:

```
static
```

Template:

```
reassemble_tm_frames(Frames,SecondaryHeaderLength,Policy,State,Packets,
UpdatedState)
```

Mode and number of proofs:

```
reassemble_tm_frames(+list(compound),+integer,+atom,+compound,-list(compound),
-compound) - one_or_error
```

Exceptions:

`Frames`, `SecondaryHeaderLength`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`Policy` is neither a variable nor a valid discontinuity recovery policy:

```
domain_error(ccsds_discontinuity_policy,Policy)
```

`State` is neither a variable nor a valid channel reassembly state term:

```
domain_error(ccsds_channel_reassembly_state,State)
```

An element `Frame` of `Frames` is neither a variable nor a valid TM transfer frame term:

```
domain_error(ccsds_tm_transfer_frame_term,Frame)
```

The TM frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_tm_transfer_frame_sequence,
tm_transfer_frame_sequence(SpacecraftId,VirtualChannelId,ExpectedFrameCount,
VirtualChannelFrameCount))
```

The stored and current TM frame counter moduli differ:

```
domain_error(ccsds_tm_transfer_frame_counter_modulus,
tm_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

reassemble_tm_frames/7

Reassembles complete packets across a sequence of TM transfer frames using the selected discontinuity recovery policy and returns any recovery events in frame order.

Compilation flags:

```
static
```

Template:

```
reassemble_tm_frames(Frames,SecondaryHeaderLength,Policy,State,Packets,
UpdatedState,Events)
```

Mode and number of proofs:

```
reassemble_tm_frames(+list(compound),+integer,+atom,+compound,-list(compound),
-compound,-list(compound)) - one_or_error
```

Exceptions:

`Frames`, `SecondaryHeaderLength`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

`Policy` is neither a variable nor a valid discontinuity recovery policy:

`domain_error(ccsds_discontinuity_policy,Policy)`

`State` is neither a variable nor a valid channel reassembly state term:

`domain_error(ccsds_channel_reassembly_state,State)`

An element `Frame` of `Frames` is neither a variable nor a valid TM transfer frame term:

`domain_error(ccsds_tm_transfer_frame_term,Frame)`

The TM frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_tm_transfer_frame_sequence,
tm_transfer_frame_sequence(SpacecraftId,VirtualChannelId,ExpectedFrameCount,
VirtualChannelFrameCount))
```

The stored and current TM frame counter moduli differ:

```
domain_error(ccsds_tm_transfer_frame_counter_modulus,
tm_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

reassemble_aos_packets/5

Extracts the packet zone from an AOS transfer frame, using the default `throw` discontinuity recovery policy, and returns the complete packets plus the updated channel reassembly state.

Compilation flags:

`static`

Template:

`reassemble_aos_packets(Frame,SecondaryHeaderLength,State,Packets,UpdatedState)`

Mode and number of proofs:

`reassemble_aos_packets(+compound,+integer,+compound,-list(compound),-compound)` - `one_or_error`

Exceptions:

`Frame`, `SecondaryHeaderLength`, or `State` is a variable:

`instantiation_error`

`Frame` is neither a variable nor a valid AOS transfer frame term:

`domain_error(ccsds_aos_transfer_frame_term,Frame)`

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

`State` is neither a variable nor a valid channel reassembly state term:

`domain_error(ccsds_channel_reassembly_state,State)`

The AOS frame sequence is discontinuous:

```
domain_error(ccsds_aos_transfer_frame_sequence,
aos_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedFrameCount,VirtualChannelFrameCount))
```

The stored and current AOS frame counter moduli differ:

```
domain_error(ccsds_aos_transfer_frame_counter_modulus,
aos_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

reassemble_aos_packets/6

Extracts the packet zone from an AOS transfer frame, applies the selected discontinuity recovery policy when frame-count continuity is broken, and returns the complete packets plus the updated channel reassembly state.

Compilation flags:

```
static
```

Template:

```
reassemble_aos_packets(Frame,SecondaryHeaderLength,Policy,State,Packets,
UpdatedState)
```

Mode and number of proofs:

```
reassemble_aos_packets(+compound,+integer,+atom,+compound,-list(compound),
-compound) - one_or_error
```

Exceptions:

`Frame`, `SecondaryHeaderLength`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`Frame` is neither a variable nor a valid AOS transfer frame term:

```
domain_error(ccsds_aos_transfer_frame_term,Frame)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`Policy` is neither a variable nor a valid discontinuity recovery policy:

```
domain_error(ccsds_discontinuity_policy,Policy)
```

`State` is neither a variable nor a valid channel reassembly state term:

```
domain_error(ccsds_channel_reassembly_state,State)
```

The AOS frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_aos_transfer_frame_sequence,
aos_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedFrameCount,VirtualChannelFrameCount))
```

The stored and current AOS frame counter moduli differ:

```
domain_error(ccsds_aos_transfer_frame_counter_modulus,
aos_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

reassemble_aos_packets/7

Extracts the packet zone from an AOS transfer frame, applies the selected discontinuity recovery policy when frame-count continuity is broken, and returns the complete packets, updated channel reassembly state, and explicit recovery events.

Compilation flags:

```
static
```

Template:

```
reassemble_aos_packets(Frame,SecondaryHeaderLength,Policy,State,Packets,
UpdatedState,Events)
```

Mode and number of proofs:

```
reassemble_aos_packets(+compound,+integer,+atom,+compound,-list(compound),
-compound,-list(compound)) - one_or_error
```

Exceptions:

`Frame`, `SecondaryHeaderLength`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`Frame` is neither a variable nor a valid AOS transfer frame term:

```
domain_error(ccsds_aos_transfer_frame_term,Frame)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`Policy` is neither a variable nor a valid discontinuity recovery policy:

```
domain_error(ccsds_discontinuity_policy,Policy)
```

`State` is neither a variable nor a valid channel reassembly state term:

```
domain_error(ccsds_channel_reassembly_state,State)
```

The AOS frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_aos_transfer_frame_sequence,
aos_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedFrameCount,VirtualChannelFrameCount))
```

The stored and current AOS frame counter moduli differ:

```
domain_error(ccsds_aos_transfer_frame_counter_modulus,
aos_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

reassemble_aos_frames/5

Reassembles complete packets across a sequence of AOS transfer frames using the default `throw` discontinuity recovery policy.

Compilation flags:

```
static
```

Template:

```
reassemble_aos_frames(Frames,SecondaryHeaderLength,State,Packets,UpdatedState)
```

Mode and number of proofs:

```
reassemble_aos_frames(+list(compound),+integer,+compound,-list(compound),
-compound) - one_or_error
```

Exceptions:

`Frames`, `SecondaryHeaderLength`, or `State` is a variable:

```
instantiation_error
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`State` is neither a variable nor a valid channel reassembly state term:

```
domain_error(ccsds_channel_reassembly_state,State)
```

An element `Frame` of `Frames` is neither a variable nor a valid AOS transfer frame term:

```
domain_error(ccsds_aos_transfer_frame_term,Frame)
```

The AOS frame sequence is discontinuous:

```
domain_error(ccsds_aos_transfer_frame_sequence,
aos_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedFrameCount,VirtualChannelFrameCount))
```

The stored and current AOS frame counter moduli differ:

```
domain_error(ccsds_aos_transfer_frame_counter_modulus,
aos_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

reassemble_aos_frames/6

Reassembles complete packets across a sequence of AOS transfer frames using the selected discontinuity recovery policy.

Compilation flags:

```
static
```

Template:

```
reassemble_aos_frames(Frames,SecondaryHeaderLength,Policy,State,Packets,
UpdatedState)
```

Mode and number of proofs:

`reassemble_aos_frames(+list(compound),+integer,+atom,+compound,-list(compound),-compound)` - `one_or_error`

Exceptions:

`Frames`, `SecondaryHeaderLength`, `Policy`, or `State` is a variable:

`instantiation_error`

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

`Policy` is neither a variable nor a valid discontinuity recovery policy:

`domain_error(ccsds_discontinuity_policy,Policy)`

`State` is neither a variable nor a valid channel reassembly state term:

`domain_error(ccsds_channel_reassembly_state,State)`

An element `Frame` of `Frames` is neither a variable nor a valid AOS transfer frame term:

`domain_error(ccsds_aos_transfer_frame_term,Frame)`

The AOS frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_aos_transfer_frame_sequence,
aos_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedFrameCount,VirtualChannelFrameCount))
```

The stored and current AOS frame counter moduli differ:

```
domain_error(ccsds_aos_transfer_frame_counter_modulus,
aos_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

---

reassemble_aos_frames/7

Reassembles complete packets across a sequence of AOS transfer frames using the selected discontinuity recovery policy and returns any recovery events in frame order.

Compilation flags:

`static`

Template:

```
reassemble_aos_frames(Frames,SecondaryHeaderLength,Policy,State,Packets,
UpdatedState,Events)
```

Mode and number of proofs:

`reassemble_aos_frames(+list(compound),+integer,+atom,+compound,-list(compound),-compound,-list(compound))` - `one_or_error`

Exceptions:

`Frames`, `SecondaryHeaderLength`, `Policy`, or `State` is a variable:

`instantiation_error`

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

`Policy` is neither a variable nor a valid discontinuity recovery policy:

`domain_error(ccsds_discontinuity_policy,Policy)`

`State` is neither a variable nor a valid channel reassembly state term:

`domain_error(ccsds_channel_reassembly_state,State)`

An element `Frame` of `Frames` is neither a variable nor a valid AOS transfer frame term:

`domain_error(ccsds_aos_transfer_frame_term,Frame)`

The AOS frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_aos_transfer_frame_sequence,
aos_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedFrameCount,VirtualChannelFrameCount))
```

The stored and current AOS frame counter moduli differ:

```
domain_error(ccsds_aos_transfer_frame_counter_modulus,
aos_transfer_frame_counter_modulus(SpacecraftId,VirtualChannelId,
StoredCounterModulus,CounterModulus))
```

A pending fragment cannot be continued by the packet zone prefix data:

```
domain_error(ccsds_packet_reassembly_continuation,
packet_reassembly_state(PendingData))
```

A continuation fragment is followed by a new packet before the previous packet is completed:

```
domain_error(ccsds_packet_reassembly_continuation,
incomplete_continuation_before_new_packet)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.34 ccsds_packetization

object

1.34.1 `ccsds_packetization`

Helpers for packetizing CCSDS space packets into TM and AOS service-data regions with cross-frame carryover and idle fill generation.

Availability:

`logtalk_load(ccsds_packetization(loader))`

Author: Paulo Moura
Version: 1:0:2
Date: 2026-06-16

Compilation flags:
`static, context_switching_calls`

Uses:
ccsds_packets(SecondaryHeaderLength)
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *initial_state/1*
  - *pending_packets/2*
  - *packetize_tm_packets/6*
  - *packetize_tm_packets/7*
  - *packetize_aos_packets/6*
  - *packetize_aos_packets/7*
  - *packetize_tm_frames/7*
  - *packetize_tm_frames/8*
  - *packetize_aos_frames/7*
  - *packetize_aos_frames/8*
  - *generate_idle_packet/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

initial_state/1

Returns the initial packetizer state.

Compilation flags:
`static`

Template:
`initial_state(State)`
Mode and number of proofs:
`initial_state(-compound)` - `one`

pending_packets/2

Extracts the non-empty queued packets and pending trailing packet bytes buffered per frame type, spacecraft identifier, and virtual channel identifier.

Compilation flags:

`static`

Template:

`pending_packets(State,PendingPackets)`

Mode and number of proofs:

`pending_packets(+compound,-list(compound))` - `one_or_error`

Exceptions:

`State` is a variable:

`instantiation_error`

`State` is neither a variable nor a valid packetizer state term:

`domain_error(ccsds_packetizer_state,State)`

---

packetize_tm_packets/6

Packetizes packets into the packet service region of a TM transfer frame, preserving any still-pending trailing packet bytes and queuing packets that do not fully fit.

Compilation flags:

`static`

Template:

`packetize_tm_packets(Frame,SecondaryHeaderLength,State,Packets,UpdatedFrame,UpdatedState)`

Mode and number of proofs:

`packetize_tm_packets(+compound,+integer,+compound,+list(compound),-compound,-compound)` - `one_or_error`

Exceptions:

`Frame`, `SecondaryHeaderLength`, `State`, or `Packets` is a variable:

`instantiation_error`

`Frame` is neither a variable nor a valid TM transfer frame term:

`domain_error(ccsds_tm_transfer_frame_term,Frame)`

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

`State` is neither a variable nor a valid packetizer state term:

`domain_error(ccsds_packetizer_state,State)`

`Packets` is neither a variable nor a valid list of CCSDS packet terms:

`domain_error(ccsds_packet_terms,Packets)`

---

packetize_tm_packets/7

Packetizes packets into the packet service region of a TM transfer frame and also returns explicit packetization events.

Compilation flags:

```
static
```

Template:

```
packetize_tm_packets(Frame,SecondaryHeaderLength,State,Packets,UpdatedFrame,
UpdatedState,Events)
```

Mode and number of proofs:

```
packetize_tm_packets(+compound,+integer,+compound,+list(compound),-compound,
-compound,-list(compound)) - one_or_error
```

Exceptions:

`Frame`, `SecondaryHeaderLength`, `State`, or `Packets` is a variable:

```
instantiation_error
```

`Frame` is neither a variable nor a valid TM transfer frame term:

```
domain_error(ccsds_tm_transfer_frame_term,Frame)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`State` is neither a variable nor a valid packetizer state term:

```
domain_error(ccsds_packetizer_state,State)
```

`Packets` is neither a variable nor a valid list of CCSDS packet terms:

```
domain_error(ccsds_packet_terms,Packets)
```

---

packetize_aos_packets/6

Packetizes packets into the packet service region of an AOS transfer frame, preserving any still-pending trailing packet bytes and queuing packets that do not fully fit.

Compilation flags:

```
static
```

Template:

```
packetize_aos_packets(Frame,SecondaryHeaderLength,State,Packets,UpdatedFrame,
UpdatedState)
```

Mode and number of proofs:

```
packetize_aos_packets(+compound,+integer,+compound,+list(compound),-compound,
-compound) - one_or_error
```

Exceptions:

`Frame`, `SecondaryHeaderLength`, `State`, or `Packets` is a variable:

```
instantiation_error
```

`Frame` is neither a variable nor a valid AOS transfer frame term:

```
domain_error(ccsds_aos_transfer_frame_term,Frame)
```

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

```
domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)
```

`State` is neither a variable nor a valid packetizer state term:
`domain_error(ccsds_packetizer_state,State)`
`Packets` is neither a variable nor a valid list of CCSDS packet terms:
`domain_error(ccsds_packet_terms,Packets)`

---

packetize_aos_packets/7

Packetizes packets into the packet service region of an AOS transfer frame and also returns explicit packetization events.

Compilation flags:
`static`

Template:
`packetize_aos_packets(Frame,SecondaryHeaderLength,State,Packets,UpdatedFrame,UpdatedState,Events)`
Mode and number of proofs:
`packetize_aos_packets(+compound,+integer,+compound,+list(compound),-compound,-compound,-list(compound))` - `one_or_error`

Exceptions:
`Frame`, `SecondaryHeaderLength`, `State`, or `Packets` is a variable:
`instantiation_error`
`Frame` is neither a variable nor a valid AOS transfer frame term:
`domain_error(ccsds_aos_transfer_frame_term,Frame)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`
`State` is neither a variable nor a valid packetizer state term:
`domain_error(ccsds_packetizer_state,State)`
`Packets` is neither a variable nor a valid list of CCSDS packet terms:
`domain_error(ccsds_packet_terms,Packets)`

---

packetize_tm_frames/7

Packetizes packets across a sequence of TM transfer frames. Packets that remain queued for the first frame channel are also returned.

Compilation flags:
`static`

Template:
`packetize_tm_frames(Frames,SecondaryHeaderLength,State,Packets,UpdatedFrames,RemainingPackets,UpdatedState)`
Mode and number of proofs:
`packetize_tm_frames(+list(compound),+integer,+compound,+list(compound),-list(compound),-list(compound),-compound)` - `one_or_error`

Exceptions:

`Frames`, `SecondaryHeaderLength`, `State`, or `Packets` is a variable:
`instantiation_error`
`Frames` is neither a variable nor a valid list of TM transfer frame terms:
`domain_error(ccsds_tm_transfer_frame_terms,Frames)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`
`State` is neither a variable nor a valid packetizer state term:
`domain_error(ccsds_packetizer_state,State)`
`Packets` is neither a variable nor a valid list of CCSDS packet terms:
`domain_error(ccsds_packet_terms,Packets)`

---

packetize_tm_frames/8

Packetizes packets across a sequence of TM transfer frames and also returns explicit packetization events in frame order.

Compilation flags:
`static`

Template:
`packetize_tm_frames(Frames,SecondaryHeaderLength,State,Packets,UpdatedFrames,RemainingPackets,UpdatedState,Events)`

Mode and number of proofs:
`packetize_tm_frames(+list(compound),+integer,+compound,+list(compound),-list(compound),-list(compound),-compound,-list(compound))` - `one_or_error`

Exceptions:
`Frames`, `SecondaryHeaderLength`, `State`, or `Packets` is a variable:
`instantiation_error`
`Frames` is neither a variable nor a valid list of TM transfer frame terms:
`domain_error(ccsds_tm_transfer_frame_terms,Frames)`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`
`State` is neither a variable nor a valid packetizer state term:
`domain_error(ccsds_packetizer_state,State)`
`Packets` is neither a variable nor a valid list of CCSDS packet terms:
`domain_error(ccsds_packet_terms,Packets)`

---

packetize_aos_frames/7

Packetizes packets across a sequence of AOS transfer frames. Packets that remain queued for the first frame channel are also returned.

Compilation flags:

`static`

Template:

`packetize_aos_frames(Frames,SecondaryHeaderLength,State,Packets,UpdatedFrames,RemainingPackets,UpdatedState)`

Mode and number of proofs:

`packetize_aos_frames(+list(compound),+integer,+compound,+list(compound),-list(compound),-list(compound),-compound)` - `one_or_error`

Exceptions:

`Frames`, `SecondaryHeaderLength`, `State`, or `Packets` is a variable:

`instantiation_error`

`Frames` is neither a variable nor a valid list of AOS transfer frame terms:

`domain_error(ccsds_aos_transfer_frame_terms,Frames)`

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

`State` is neither a variable nor a valid packetizer state term:

`domain_error(ccsds_packetizer_state,State)`

`Packets` is neither a variable nor a valid list of CCSDS packet terms:

`domain_error(ccsds_packet_terms,Packets)`

---

packetize_aos_frames/8

Packetizes packets across a sequence of AOS transfer frames and also returns explicit packetization events in frame order.

Compilation flags:

`static`

Template:

`packetize_aos_frames(Frames,SecondaryHeaderLength,State,Packets,UpdatedFrames,RemainingPackets,UpdatedState,Events)`

Mode and number of proofs:

`packetize_aos_frames(+list(compound),+integer,+compound,+list(compound),-list(compound),-list(compound),-compound,-list(compound))` - `one_or_error`

Exceptions:

`Frames`, `SecondaryHeaderLength`, `State`, or `Packets` is a variable:

`instantiation_error`

`Frames` is neither a variable nor a valid list of AOS transfer frame terms:

`domain_error(ccsds_aos_transfer_frame_terms,Frames)`

`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:

`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`

`State` is neither a variable nor a valid packetizer state term:
`domain_error(ccsds_packetizer_state,State)`
`Packets` is neither a variable nor a valid list of CCSDS packet terms:
`domain_error(ccsds_packet_terms,Packets)`

`generate_idle_packet/4`

Generates a telemetry idle packet using APID 2047, the given packet secondary header length, sequence count, and user-data length.

Compilation flags:
`static`

Template:
`generate_idle_packet(SecondaryHeaderLength,SequenceCount,UserDataLength,Packet)`
Mode and number of proofs:
`generate_idle_packet(+integer,+integer,+integer,-compound)` - `one_or_error`

Exceptions:
`SecondaryHeaderLength`, `SequenceCount`, or `UserDataLength` is a variable:
`instantiation_error`
`SecondaryHeaderLength` is neither a variable nor a valid CCSDS packet secondary header length:
`domain_error(ccsds_secondary_header_length,SecondaryHeaderLength)`
`SequenceCount` is neither a variable nor a valid CCSDS packet sequence count:
`domain_error(ccsds_packet_sequence_count,SequenceCount)`
`UserDataLength` is neither a variable nor a valid idle-packet user-data length:
`domain_error(ccsds_idle_user_data_length,UserDataLength)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.35 ccsds_packets

object

1.35.1 `ccsds_packets`

CCSDS Space Packet parser with no secondary header parsing. For secondary header support, use ccsds_packets(Length) where Length is the secondary header size in bytes.

Availability:
`logtalk_load(ccsds_packets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2025-12-05

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *ccsds_packets(0)*

Remarks:
(none)

Inherited public predicates:
apid/2 *data_length/2* *generate/2* *generate/3* *parse/2* *secondary_header/2* *secondary_header_flag/2* *secondary_header_time/3* *sequence_count/2* *sequence_flags/2* *type/2* *user_data/2* *version/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.35.2 `ccsds_packets(SecondaryHeaderLength)`

- `SecondaryHeaderLength` - Length in bytes of the secondary header when present (0 for no secondary header parsing, or a positive integer).

CCSDS Space Packet parser following the CCSDS 133.0-B-2 standard. Parses binary packet data including optional secondary headers.

Availability:
`logtalk_load(ccsds_packets(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-06-24

Compilation flags:
`static, context_switching_calls`

Uses:
ccsds_time_fields
list
reader

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
  - *generate/3*
  - *version/2*
  - *type/2*
  - *secondary_header_flag/2*
  - *apid/2*
  - *sequence_flags/2*
  - *sequence_count/2*
  - *data_length/2*

  - *user_data/2*
  - *secondary_header/2*
  - *secondary_header_time/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses CCSDS packet(s) from a source into a list of packet terms. The source can be `file(File)`, `stream(Stream)`, or `bytes(Bytes)`.

Compilation flags:
`static`

Template:
`parse(Source,Packets)`
Mode and number of proofs:
`parse(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(ccsds_source,Source)`
`Source` is a valid source but the data cannot be parsed as a CCSDS packet:
`domain_error(ccsds_byte_sequence,Bytes)`

---

generate/2

Generates CCSDS packet bytes to a sink from a list of packet terms. The sink can be `file(File)`, `stream(Stream)`, or `bytes(Bytes)`. For `file(File)` and `stream(Stream)`, writes to the binary file or stream. For `bytes(Bytes)`, unifies Bytes with the generated byte list.

Compilation flags:
`static`

Template:
`generate(Sink,Packets)`
Mode and number of proofs:
`generate(+compound,+list(compound))` - `one_or_error`

Exceptions:
`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid sink:

`domain_error(ccsds_sink,Sink)`

`Packets` is a partial list or a list with an element `Packet` which is a variable:

`instantiation_error`

An element `Packet` of the list `Packets` is neither a variable nor a valid CCSDS packet term:

`domain_error(ccsds_packet_term,Packet)`

---

generate/3

Generates a list of bytes from a CCSDS packet term with an open tail. Mainly used when generating arbitrary CCSDS packets.

Compilation flags:

`static`

Template:

`generate(Packet,Bytes,Tail)`

Mode and number of proofs:

`generate(+compound,-list(byte),--variable)` - `one_or_error`

Exceptions:

`Packet` is a variable:

`instantiation_error`

`Packet` is neither a variable nor a valid CCSDS packet term:

`domain_error(ccsds_packet_term,Packet)`

---

version/2

Extracts the version number from a packet (always 0 for CCSDS Space Packets).

Compilation flags:

`static`

Template:

`version(Packet,Version)`

Mode and number of proofs:

`version(+compound,-integer)` - `one`

---

type/2

Extracts the packet type from a packet. Returns `telemetry` or `telecommand`.

Compilation flags:
`static`

Template:
`type(Packet,Type)`
Mode and number of proofs:
`type(+compound,-atom)` - `one`

---

secondary_header_flag/2

Extracts the secondary header flag. Returns `absent` or `present`.

Compilation flags:
`static`

Template:
`secondary_header_flag(Packet,Flag)`
Mode and number of proofs:
`secondary_header_flag(+compound,-atom)` - `one`

---

apid/2

Extracts the Application Process Identifier (APID) from a packet.

Compilation flags:
`static`

Template:
`apid(Packet,APID)`
Mode and number of proofs:
`apid(+compound,-integer)` - `one`

---

sequence_flags/2

Extracts the sequence flags. Returns `continuation`, `first`, `last`, or `standalone`.

Compilation flags:
`static`

Template:
`sequence_flags(Packet,Flags)`
Mode and number of proofs:
`sequence_flags(+compound,-atom)` - `one`

---

sequence_count/2

Extracts the packet sequence count (0-16383).

Compilation flags:
`static`

Template:
`sequence_count(Packet,Count)`
Mode and number of proofs:
`sequence_count(+compound,-integer)` - `one`

---

data_length/2

Extracts the packet data length field value.

Compilation flags:
`static`

Template:
`data_length(Packet,Length)`
Mode and number of proofs:
`data_length(+compound,-integer)` - `one`

---

user_data/2

Extracts the user data field as a list of bytes.

Compilation flags:
`static`

Template:
`user_data(Packet,Data)`
Mode and number of proofs:
`user_data(+compound,-list(byte))` - `one`

---

secondary_header/2

Extracts the secondary header. Returns none if not present, or secondary_header(Bytes) with the raw bytes.

Compilation flags:
`static`

Template:
`secondary_header(Packet,SecondaryHeader)`
Mode and number of proofs:
`secondary_header(+compound,-compound)` - `one`

---

secondary_header_time/3

Parses time from a packet secondary header. When `Descriptor` is a variable, the secondary header is parsed as a self-describing CCSDS time field. When `Descriptor` is instantiated, it is used to parse the raw T-field bytes.

Compilation flags:
`static`

Template:
`secondary_header_time(Packet,Descriptor,TimeCode)`
Mode and number of proofs:
`secondary_header_time(+compound,?compound,-compound)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.35.3 `ccsds_packets_types`

Type definitions and arbitrary generators for CCSDS packets.

Availability:
`logtalk_load(ccsds_packets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-08

Compilation flags:
`static`

Provides:
type::type/1
type::check/2
arbitrary::arbitrary/1
arbitrary::arbitrary/2
Uses:
ccsds_packets(SecondaryHeaderLength)
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.36 ccsds_tc_services

object

1.36.1 `ccsds_tc_services`

Helpers for CCSDS telecommand segment extraction, insertion, and cross-frame TC service reassembly.

Availability:
`logtalk_load(ccsds_tc_services(loader))`

Author: Paulo Moura
Version: 0:1:1
Date: 2026-06-14

Compilation flags:
`static, context_switching_calls`

Uses:
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *valid_segment/1*
  - *sequence_flags/2*
  - *map_id/2*
  - *valid_map_dispatch/1*

- *dispatch_map_id/2*
- *dispatch_service_units/2*
- *segment_header_suffix/2*
- *valid_reassembled_segment/1*
- *segment_header_suffixes/2*
- *segment_data/2*
- *valid_reassembly_state/1*
- *initial_reassembly_state/1*
- *pending_fragments/2*
- *valid_discontinuity_policy/1*
- *extract_tc_segment/2*
- *insert_tc_segment/3*
- *reassemble_tc_frame/4*
- *reassemble_tc_frame/5*
- *reassemble_tc_frame/6*
- *reassemble_tc_frames/4*
- *reassemble_tc_frames/5*
- *reassemble_tc_frames/6*
- *reassemble_tc_frame_with_provenance/5*
- *reassemble_tc_frame_with_provenance/6*
- *reassemble_tc_frames_with_provenance/5*
- *reassemble_tc_frames_with_provenance/6*
- *dispatch_service_units_by_map/2*
- *dispatch_service_units_by_map/3*

• *Protected predicates*
• *Private predicates*
• *Operators*

Public predicates

valid_segment/1

True if the argument is a valid CCSDS telecommand segment term.

Compilation flags:
`static`

Template:
`valid_segment(Segment)`

Mode and number of proofs:
`valid_segment(@compound)` - `zero_or_one`

sequence_flags/2

Extracts the telecommand segment sequence flags.

Compilation flags:
`static`

Template:
`sequence_flags(Segment,SequenceFlags)`
Mode and number of proofs:
`sequence_flags(+compound,-atom)` - `one`

---

map_id/2

Extracts the telecommand service-unit MAP identifier from either a telecommand segment term or a provenance-aware reassembled telecommand service-unit term.

Compilation flags:
`static`

Template:
`map_id(ServiceUnit,MapId)`
Mode and number of proofs:
`map_id(+compound,-integer)` - `one`

---

valid_map_dispatch/1

True if the argument is a valid MAP-aware dispatch term grouping complete telecommand service units for a single MAP identifier.

Compilation flags:
`static`

Template:
`valid_map_dispatch(Dispatch)`
Mode and number of proofs:
`valid_map_dispatch(@compound)` - `zero_or_one`

---

dispatch_map_id/2

Extracts the MAP identifier from a MAP-aware dispatch term.

Compilation flags:
```
static
```

Template:
```
dispatch_map_id(Dispatch,MapId)
```
Mode and number of proofs:
```
dispatch_map_id(+compound,-integer) - one
```

---

dispatch_service_units/2

Extracts the complete telecommand service units from a MAP-aware dispatch term.

Compilation flags:
```
static
```

Template:
```
dispatch_service_units(Dispatch,ServiceUnits)
```
Mode and number of proofs:
```
dispatch_service_units(+compound,-list(compound)) - one
```

---

segment_header_suffix/2

Extracts the mission-specific segment-header suffix bytes. Returns the empty list when the segment has no mission-specific suffix.

Compilation flags:
```
static
```

Template:
```
segment_header_suffix(Segment,HeaderSuffix)
```
Mode and number of proofs:
```
segment_header_suffix(+compound,-list(byte)) - one
```

---

valid_reassembled_segment/1

True if the argument is a valid provenance-aware reassembled telecommand service-unit term.

Compilation flags:

```
static
```

Template:

```
valid_reassembled_segment(ReassembledSegment)
```

Mode and number of proofs:

```
valid_reassembled_segment(@compound) - zero_or_one
```

---

segment_header_suffixes/2

Extracts the ordered list of per-frame mission-specific segment-header suffixes that contributed to a provenance-aware reassembled telecommand service-unit term.

Compilation flags:

```
static
```

Template:

```
segment_header_suffixes(ReassembledSegment,HeaderSuffixes)
```

Mode and number of proofs:

```
segment_header_suffixes(+compound,-list(list(byte))) - one
```

---

segment_data/2

Extracts the telecommand segment data bytes.

Compilation flags:

```
static
```

Template:

```
segment_data(Segment,Data)
```

Mode and number of proofs:

```
segment_data(+compound,-list(byte)) - one
```

---

valid_reassembly_state/1

True if the argument is a valid telecommand segment reassembly state term.

Compilation flags:
`static`

Template:
`valid_reassembly_state(State)`
Mode and number of proofs:
`valid_reassembly_state(@compound)` - `zero_or_one`

---

initial_reassembly_state/1

Returns the initial telecommand segment reassembly state.

Compilation flags:
`static`

Template:
`initial_reassembly_state(State)`
Mode and number of proofs:
`initial_reassembly_state(-compound)` - `one`

---

pending_fragments/2

Extracts the pending segmented telecommand data fragments keyed by spacecraft, virtual channel, and MAP identifier.

Compilation flags:
`static`

Template:
`pending_fragments(State,PendingFragments)`
Mode and number of proofs:
`pending_fragments(+compound,-list(compound))` - `one`

---

valid_discontinuity_policy/1

True if the argument is a valid discontinuity recovery policy atom. Valid values are `throw`, `drop`, and `resynchronize`.

Compilation flags:
`static`

Template:
`valid_discontinuity_policy(Policy)`
Mode and number of proofs:
`valid_discontinuity_policy(@atom)` - `zero_or_one`

---

extract_tc_segment/2

Extracts a telecommand segment from a telecommand transfer frame using the representation `tc_segment(SequenceFlags, MapId, HeaderSuffix, Data)`. When mission-specific segment headers use multiple octets, only the first octet is interpreted for the standard MAP and sequence semantics and the remaining octets are returned as `HeaderSuffix`.

Compilation flags:
`static`

Template:
`extract_tc_segment(Frame,Segment)`
Mode and number of proofs:
`extract_tc_segment(+compound,-compound)` - `one_or_error`

Exceptions:
`Frame` is a variable:
`instantiation_error`
`Frame` is neither a variable nor a valid telecommand transfer frame term:
`domain_error(ccsds_tc_transfer_frame_term,Frame)`

---

insert_tc_segment/3

Updates a telecommand transfer frame from a telecommand segment term by encoding the standard MAP and sequence semantics in the first segment-header octet and preserving any remaining mission-specific segment-header octets.

Compilation flags:
`static`

Template:
`insert_tc_segment(Segment,Frame,UpdatedFrame)`
Mode and number of proofs:
`insert_tc_segment(+compound,+compound,-compound)` - `one_or_error`

Exceptions:

`Segment` or `Frame` is a variable:
`instantiation_error`
`Segment` is neither a variable nor a valid telecommand segment term:
`domain_error(ccsds_tc_segment,Segment)`
`Frame` has an invalid telecommand segment header suffix encoding:
`domain_error(ccsds_tc_segment_header,SegmentHeader)`

---

reassemble_tc_frame/4

Reassembles complete telecommand service units from a single telecommand transfer frame using the default `throw` discontinuity recovery policy.

Compilation flags:
`static`

Template:
`reassemble_tc_frame(Frame,State,Segments,UpdatedState)`
Mode and number of proofs:
`reassemble_tc_frame(+compound,+compound,-list(compound),-compound)` - `one_or_error`

Exceptions:
`Frame` or `State` is a variable:
`instantiation_error`
`Frame` is neither a variable nor a valid telecommand transfer frame term:
`domain_error(ccsds_tc_transfer_frame_term,Frame)`
`State` is neither a variable nor a valid telecommand reassembly state term:
`domain_error(ccsds_tc_reassembly_state,State)`
The telecommand frame sequence is discontinuous:
```
domain_error(ccsds_tc_transfer_frame_sequence,
tc_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedSequenceNumber,SequenceNumber))
```
A telecommand segment starts a new reassembly while a previous segment for the same MAP is still pending:
`domain_error(ccsds_tc_segment_reassembly,existing_pending_segment(MapId))`
A telecommand continuation or last segment is missing its first segment:
`domain_error(ccsds_tc_segment_reassembly,missing_first_segment(MapId))`

---

reassemble_tc_frame/5

Reassembles complete telecommand service units from a single telecommand transfer frame using the selected discontinuity recovery policy.

Compilation flags:
`static`

Template:

```
reassemble_tc_frame(Frame,Policy,State,Segments,UpdatedState)
```

Mode and number of proofs:

```
reassemble_tc_frame(+compound,+atom,+compound,-list(compound),-compound) -
one_or_error
```

Exceptions:

`Frame`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`Frame` is neither a variable nor a valid telecommand transfer frame term:

```
domain_error(ccsds_tc_transfer_frame_term,Frame)
```

`Policy` is neither a variable nor a valid telecommand discontinuity recovery policy:

```
domain_error(ccsds_tc_discontinuity_policy,Policy)
```

`State` is neither a variable nor a valid telecommand reassembly state term:

```
domain_error(ccsds_tc_reassembly_state,State)
```

The telecommand frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_tc_transfer_frame_sequence,
tc_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedSequenceNumber,SequenceNumber))
```

A telecommand segment starts a new reassembly while a previous segment for the same MAP is still pending:

```
domain_error(ccsds_tc_segment_reassembly,existing_pending_segment(MapId))
```

A telecommand continuation or last segment is missing its first segment:

```
domain_error(ccsds_tc_segment_reassembly,missing_first_segment(MapId))
```

---

reassemble_tc_frame/6

Reassembles complete telecommand service units from a single telecommand transfer frame using the selected discontinuity recovery policy and returns any recovery events.

Compilation flags:

```
static
```

Template:

```
reassemble_tc_frame(Frame,Policy,State,Segments,UpdatedState,Events)
```

Mode and number of proofs:

```
reassemble_tc_frame(+compound,+atom,+compound,-list(compound),-compound,
-list(compound)) - one_or_error
```

Exceptions:

`Frame`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`Frame` is neither a variable nor a valid telecommand transfer frame term:

```
domain_error(ccsds_tc_transfer_frame_term,Frame)
```

`Policy` is neither a variable nor a valid telecommand discontinuity recovery policy:

```
domain_error(ccsds_tc_discontinuity_policy,Policy)
```

`State` is neither a variable nor a valid telecommand reassembly state term:

```
domain_error(ccsds_tc_reassembly_state,State)
```

The telecommand frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_tc_transfer_frame_sequence,
tc_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedSequenceNumber,SequenceNumber))
```

A telecommand segment starts a new reassembly while a previous segment for the same MAP is still pending:

```
domain_error(ccsds_tc_segment_reassembly,existing_pending_segment(MapId))
```

A telecommand continuation or last segment is missing its first segment:

```
domain_error(ccsds_tc_segment_reassembly,missing_first_segment(MapId))
```

---

reassemble_tc_frames/4

Reassembles complete telecommand service units across a sequence of telecommand transfer frames using the default `throw` discontinuity recovery policy.

Compilation flags:

```
static
```

Template:

```
reassemble_tc_frames(Frames,State,Segments,UpdatedState)
```

Mode and number of proofs:

```
reassemble_tc_frames(+list(compound),+compound,-list(compound),-compound) -
one_or_error
```

Exceptions:

`Frames` or `State` is a variable:

```
instantiation_error
```

`State` is neither a variable nor a valid telecommand reassembly state term:

```
domain_error(ccsds_tc_reassembly_state,State)
```

An element `Frame` of `Frames` is neither a variable nor a valid telecommand transfer frame term:

```
domain_error(ccsds_tc_transfer_frame_term,Frame)
```

The telecommand frame sequence is discontinuous:

```
domain_error(ccsds_tc_transfer_frame_sequence,
tc_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedSequenceNumber,SequenceNumber))
```

A telecommand segment starts a new reassembly while a previous segment for the same MAP is still pending:

```
domain_error(ccsds_tc_segment_reassembly,existing_pending_segment(MapId))
```

A telecommand continuation or last segment is missing its first segment:

```
domain_error(ccsds_tc_segment_reassembly,missing_first_segment(MapId))
```

---

reassemble_tc_frames/5

Reassembles complete telecommand service units across a sequence of telecommand transfer frames using the selected discontinuity recovery policy.

Compilation flags:

```
static
```

Template:

```
reassemble_tc_frames(Frames,Policy,State,Segments,UpdatedState)
```

Mode and number of proofs:

```
reassemble_tc_frames(+list(compound),+atom,+compound,-list(compound),-compound) -
one_or_error
```

Exceptions:

`Frames`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`Policy` is neither a variable nor a valid telecommand discontinuity recovery policy:

```
domain_error(ccsds_tc_discontinuity_policy,Policy)
```

`State` is neither a variable nor a valid telecommand reassembly state term:

```
domain_error(ccsds_tc_reassembly_state,State)
```

An element `Frame` of `Frames` is neither a variable nor a valid telecommand transfer frame term:

```
domain_error(ccsds_tc_transfer_frame_term,Frame)
```

The telecommand frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_tc_transfer_frame_sequence,
tc_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedSequenceNumber,SequenceNumber))
```

A telecommand segment starts a new reassembly while a previous segment for the same MAP is still pending:

```
domain_error(ccsds_tc_segment_reassembly,existing_pending_segment(MapId))
```

A telecommand continuation or last segment is missing its first segment:

```
domain_error(ccsds_tc_segment_reassembly,missing_first_segment(MapId))
```

---

reassemble_tc_frames/6

Reassembles complete telecommand service units across a sequence of telecommand transfer frames using the selected discontinuity recovery policy and returns any recovery events in frame order.

Compilation flags:

```
static
```

Template:

```
reassemble_tc_frames(Frames,Policy,State,Segments,UpdatedState,Events)
```

Mode and number of proofs:

```
reassemble_tc_frames(+list(compound),+atom,+compound,-list(compound),-compound,
-list(compound)) - one_or_error
```

Exceptions:

`Frames`, `Policy`, or `State` is a variable:

instantiation_error

Policy is neither a variable nor a valid telecommand discontinuity recovery policy:

domain_error(ccsds_tc_discontinuity_policy,Policy)

State is neither a variable nor a valid telecommand reassembly state term:

domain_error(ccsds_tc_reassembly_state,State)

An element Frame of Frames is neither a variable nor a valid telecommand transfer frame term:

domain_error(ccsds_tc_transfer_frame_term,Frame)

The telecommand frame sequence is discontinuous under the throw recovery policy:

```
domain_error(ccsds_tc_transfer_frame_sequence,
tc_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedSequenceNumber,SequenceNumber))
```

A telecommand segment starts a new reassembly while a previous segment for the same MAP is still pending:

domain_error(ccsds_tc_segment_reassembly,existing_pending_segment(MapId))

A telecommand continuation or last segment is missing its first segment:

domain_error(ccsds_tc_segment_reassembly,missing_first_segment(MapId))

---

reassemble_tc_frame_with_provenance/5

Reassembles complete telecommand service units from a single telecommand transfer frame using the default throw discontinuity recovery policy and returns provenance-aware reassembled service-unit terms.

Compilation flags:

static

Template:

```
reassemble_tc_frame_with_provenance(Frame,State,ReassembledSegments,UpdatedState,
Events)
```

Mode and number of proofs:

```
reassemble_tc_frame_with_provenance(+compound,+compound,-list(compound),
-compound,-list(compound)) - one_or_error
```

Exceptions:

Frame or State is a variable:

instantiation_error

Frame is neither a variable nor a valid telecommand transfer frame term:

domain_error(ccsds_tc_transfer_frame_term,Frame)

State is neither a variable nor a valid telecommand reassembly state term:

domain_error(ccsds_tc_reassembly_state,State)

The telecommand frame sequence is discontinuous:

```
domain_error(ccsds_tc_transfer_frame_sequence,
tc_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedSequenceNumber,SequenceNumber))
```

A telecommand segment starts a new reassembly while a previous segment for the same MAP is still pending:

domain_error(ccsds_tc_segment_reassembly,existing_pending_segment(MapId))

A telecommand continuation or last segment is missing its first segment:

domain_error(ccsds_tc_segment_reassembly,missing_first_segment(MapId))

---

reassemble_tc_frame_with_provenance/6

Reassembles complete telecommand service units from a single telecommand transfer frame using the selected discontinuity recovery policy and returns provenance-aware reassembled service-unit terms plus any recovery events.

Compilation flags:

`static`

Template:

`reassemble_tc_frame_with_provenance(Frame,Policy,State,ReassembledSegments,UpdatedState,Events)`

Mode and number of proofs:

`reassemble_tc_frame_with_provenance(+compound,+atom,+compound,-list(compound),-compound,-list(compound))` - `one_or_error`

Exceptions:

`Frame`, `Policy`, or `State` is a variable:

`instantiation_error`

`Frame` is neither a variable nor a valid telecommand transfer frame term:

`domain_error(ccsds_tc_transfer_frame_term,Frame)`

`Policy` is neither a variable nor a valid telecommand discontinuity recovery policy:

`domain_error(ccsds_tc_discontinuity_policy,Policy)`

`State` is neither a variable nor a valid telecommand reassembly state term:

`domain_error(ccsds_tc_reassembly_state,State)`

The telecommand frame sequence is discontinuous under the `throw` recovery policy:

`domain_error(ccsds_tc_transfer_frame_sequence,tc_transfer_frame_sequence(SpacecraftId,VirtualChannelId,ExpectedSequenceNumber,SequenceNumber))`

A telecommand segment starts a new reassembly while a previous segment for the same MAP is still pending:

`domain_error(ccsds_tc_segment_reassembly,existing_pending_segment(MapId))`

A telecommand continuation or last segment is missing its first segment:

`domain_error(ccsds_tc_segment_reassembly,missing_first_segment(MapId))`

---

reassemble_tc_frames_with_provenance/5

Reassembles complete telecommand service units across a sequence of telecommand transfer frames using the default `throw` discontinuity recovery policy and returns provenance-aware reassembled service-unit terms.

Compilation flags:

`static`

Template:

`reassemble_tc_frames_with_provenance(Frames,State,ReassembledSegments,UpdatedState,Events)`

Mode and number of proofs:

`reassemble_tc_frames_with_provenance(+list(compound),+compound,-list(compound),-compound,-list(compound))` - `one_or_error`

Exceptions:

`Frames` or `State` is a variable:

```
instantiation_error
```

`State` is neither a variable nor a valid telecommand reassembly state term:

```
domain_error(ccsds_tc_reassembly_state,State)
```

An element `Frame` of `Frames` is neither a variable nor a valid telecommand transfer frame term:

```
domain_error(ccsds_tc_transfer_frame_term,Frame)
```

The telecommand frame sequence is discontinuous:

```
domain_error(ccsds_tc_transfer_frame_sequence,
tc_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedSequenceNumber,SequenceNumber))
```

A telecommand segment starts a new reassembly while a previous segment for the same MAP is still pending:

```
domain_error(ccsds_tc_segment_reassembly,existing_pending_segment(MapId))
```

A telecommand continuation or last segment is missing its first segment:

```
domain_error(ccsds_tc_segment_reassembly,missing_first_segment(MapId))
```

`reassemble_tc_frames_with_provenance/6`

Reassembles complete telecommand service units across a sequence of telecommand transfer frames using the selected discontinuity recovery policy and returns provenance-aware reassembled service-unit terms plus any recovery events in frame order.

Compilation flags:

```
static
```

Template:

```
reassemble_tc_frames_with_provenance(Frames,Policy,State,ReassembledSegments,
UpdatedState,Events)
```

Mode and number of proofs:

```
reassemble_tc_frames_with_provenance(+list(compound),+atom,+compound,
-list(compound),-compound,-list(compound)) - one_or_error
```

Exceptions:

`Frames`, `Policy`, or `State` is a variable:

```
instantiation_error
```

`Policy` is neither a variable nor a valid telecommand discontinuity recovery policy:

```
domain_error(ccsds_tc_discontinuity_policy,Policy)
```

`State` is neither a variable nor a valid telecommand reassembly state term:

```
domain_error(ccsds_tc_reassembly_state,State)
```

An element `Frame` of `Frames` is neither a variable nor a valid telecommand transfer frame term:

```
domain_error(ccsds_tc_transfer_frame_term,Frame)
```

The telecommand frame sequence is discontinuous under the `throw` recovery policy:

```
domain_error(ccsds_tc_transfer_frame_sequence,
tc_transfer_frame_sequence(SpacecraftId,VirtualChannelId,
ExpectedSequenceNumber,SequenceNumber))
```

A telecommand segment starts a new reassembly while a previous segment for the same MAP is still pending:

```
domain_error(ccsds_tc_segment_reassembly,existing_pending_segment(MapId))
```

A telecommand continuation or last segment is missing its first segment:

```
domain_error(ccsds_tc_segment_reassembly,missing_first_segment(MapId))
```

---

dispatch_service_units_by_map/2

Groups complete telecommand service-unit terms by MAP identifier, preserving the original service-unit order within each MAP bucket and the first-seen order of MAP buckets. Accepts both telecommand segment terms and provenance-aware reassembled telecommand service-unit terms.

Compilation flags:
`static`

Template:
`dispatch_service_units_by_map(ServiceUnits,Dispatches)`
Mode and number of proofs:
`dispatch_service_units_by_map(+list(compound),-list(compound))` - `one_or_error`

Exceptions:
`ServiceUnits` is a variable:
`instantiation_error`
`ServiceUnits` is neither a variable nor a valid list of telecommand service units:
`domain_error(ccsds_tc_service_units,ServiceUnits)`

---

dispatch_service_units_by_map/3

Extracts the complete telecommand service units for a specific MAP identifier from a list of complete telecommand service-unit terms.

Compilation flags:
`static`

Template:
`dispatch_service_units_by_map(ServiceUnits,MapId,DispatchedServiceUnits)`
Mode and number of proofs:
`dispatch_service_units_by_map(+list(compound),+integer,-list(compound))` - `one_or_error`

Exceptions:
`ServiceUnits` or `MapId` is a variable:
`instantiation_error`
`ServiceUnits` is neither a variable nor a valid list of telecommand service units:
`domain_error(ccsds_tc_service_units,ServiceUnits)`
`MapId` is neither a variable nor a valid telecommand MAP identifier:
`domain_error(ccsds_tc_map_id,MapId)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.37 ccsds_time_codes

object

1.37.1 `ccsds_ccs(CalendarVariant,FractionOctets)`

- `CalendarVariant` - Calendar representation. Supported values are `calendar` and `day_of_year`.
- `FractionOctets` - Number of BCD fraction octets. Supported values are non-negative integers.

CCSDS calendar segmented time code parser and generator.

Availability:
`logtalk_load(ccsds_time_codes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-06-25

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ccsds_time_code_protocol*
Uses:
reader
type

Remarks:
(none)

Inherited public predicates:
epoch/1 *format/1* *from_unix_seconds/2* *generate/2* *generate/3* *parse/2* *unix_seconds/2* *valid/1*

- *Public predicates*
  - *year/2*
  - *day_of_year/2*
  - *month/2*

    - *day/2*
    - *time_of_day/4*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

year/2

Extracts the year value from a CCS time term.

Compilation flags:
```
static
```

Template:
```
year(TimeCode,Year)
```
Mode and number of proofs:
```
year(+compound,-integer) - one
```

---

day_of_year/2

Extracts the day of year from an ordinal CCS time term.

Compilation flags:
```
static
```

Template:
```
day_of_year(TimeCode,DayOfYear)
```
Mode and number of proofs:
```
day_of_year(+compound,-integer) - zero_or_one
```

---

month/2

Extracts the month from a calendar CCS time term.

Compilation flags:
```
static
```

Template:
```
month(TimeCode,Month)
```
Mode and number of proofs:
```
month(+compound,-integer) - zero_or_one
```

---

day/2

Extracts the day of month from a calendar CCS time term.

Compilation flags:
`static`

Template:
`day(TimeCode,Day)`
Mode and number of proofs:
`day(+compound,-integer)` - `zero_or_one`

---

time_of_day/4

Extracts the hour, minute, and seconds-with-fraction from a CCS time term.

Compilation flags:
`static`

Template:
`time_of_day(TimeCode,Hour,Minute,Seconds)`
Mode and number of proofs:
`time_of_day(+compound,-integer,-integer,-number)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.37.2 ccsds_cds(DaySegmentOctets,SubmillisecondOctets,Epoch)

- `DaySegmentOctets` - Number of day segment octets. Supported values are `2` and `3`.
- `SubmillisecondOctets` - Number of submillisecond octets. Supported values are `0`, `2`, and `4`.
- `Epoch` - Epoch used when converting to and from Unix seconds. Supported values are `ccsds_epoch` and `unix_epoch`.

CCSDS day segmented time code parser and generator.

Availability:
`logtalk_load(ccsds_time_codes(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-06-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ccsds_time_code_protocol*
Uses:
reader
type

Remarks:
(none)

Inherited public predicates:
epoch/1 format/1 from_unix_seconds/2 generate/2 generate/3 parse/2 unix_seconds/2 valid/1

- *Public predicates*
  - *days/2*
  - *milliseconds/2*
  - *submilliseconds/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

days/2

Extracts the day segment value from a CDS time term.

Compilation flags:
`static`

Template:
`days(TimeCode,Days)`
Mode and number of proofs:
`days(+compound,-integer)` - `one`

milliseconds/2

Extracts the millisecond-of-day value from a CDS time term.

Compilation flags:
`static`

Template:
`milliseconds(TimeCode,Milliseconds)`
Mode and number of proofs:
`milliseconds(+compound,-integer)` - `one`

---

submilliseconds/2

Extracts the optional submillisecond value from a CDS time term. Two-octet objects use microseconds; four-octet objects use picoseconds within the millisecond.

Compilation flags:
`static`

Template:
`submilliseconds(TimeCode,Submilliseconds)`
Mode and number of proofs:
`submilliseconds(+compound,-integer)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.37.3 ccsds_cuc(CoarseOctets,FineOctets,Epoch)

- `CoarseOctets` - Number of coarse time octets.
- `FineOctets` - Number of fine time octets.
- `Epoch` - Epoch used when converting to and from Unix seconds. Supported values are `ccsds_epoch` and `unix_epoch`.

CCSDS unsegmented time code parser and generator.

Availability:

```
logtalk_load(ccsds_time_codes(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ccsds_time_code_protocol*
Uses:
list
reader

Remarks:
(none)

Inherited public predicates:
epoch/1 format/1 from_unix_seconds/2 generate/2 generate/3 parse/2 unix_seconds/2 valid/1

- *Public predicates*
  - *coarse/2*
  - *fine/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

coarse/2

Extracts the coarse field from a CUC time term.

Compilation flags:
`static`

Template:
`coarse(TimeCode,Coarse)`
Mode and number of proofs:
`coarse(+compound,-integer)` - `one`

`fine/2`

Extracts the fine field from a CUC time term.

Compilation flags:
`static`

Template:
`fine(TimeCode,Fine)`
Mode and number of proofs:
`fine(+compound,-integer)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.37.4 `ccsds_time_code_protocol`

CCSDS time code parser and generator protocol.

Availability:
`logtalk_load(ccsds_time_codes(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
  - *generate/3*
  - *valid/1*
  - *format/1*
  - *epoch/1*
  - *unix_seconds/2*
  - *from_unix_seconds/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a CCSDS time code from a source term. Supported source terms are `file(File)`, `stream(Stream)`, and `bytes(Bytes)`.

Compilation flags:
`static`

Template:
`parse(Source,TimeCode)`

Mode and number of proofs:
`parse(+compound,-compound)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(ccsds_time_code_source,Source)`
`Source` contents are not a valid encoding for the selected time code object:
`domain_error(ccsds_time_code_byte_sequence,Bytes)`

---

generate/2

Generates a CCSDS time code to a sink term. Supported sink terms are `file(File)`, `stream(Stream)`, and `bytes(Bytes)`.

Compilation flags:
`static`

Template:

`generate(Sink,TimeCode)`

Mode and number of proofs:

`generate(+compound,+compound)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid sink:

`domain_error(ccsds_time_code_sink,Sink)`

`TimeCode` is neither a variable nor a valid time code term for the selected object:

`domain_error(ccsds_time_code_term,TimeCode)`

`generate/3`

Generates a list of bytes from a CCSDS time code term with an open tail. Mainly used when generating arbitrary values.

Compilation flags:

`static`

Template:

`generate(TimeCode,Bytes,Tail)`

Mode and number of proofs:

`generate(+compound,-list(byte),--variable)` - `one_or_error`

Exceptions:

`TimeCode` is a variable:

`instantiation_error`

`TimeCode` is neither a variable nor a valid time code term for the selected object:

`domain_error(ccsds_time_code_term,TimeCode)`

`valid/1`

True if the argument is a valid time code term for the selected object.

Compilation flags:

`static`

Template:

`valid(TimeCode)`

Mode and number of proofs:

`valid(@compound)` - `zero_or_one`

---

format/1

Returns the CCSDS time code format handled by the object.

Compilation flags:
`static`

Template:
`format(Format)`
Mode and number of proofs:
`format(-atom)` - `one`

---

epoch/1

Returns the epoch used by the object.

Compilation flags:
`static`

Template:
`epoch(Epoch)`
Mode and number of proofs:
`epoch(-atom)` - `one`

---

unix_seconds/2

Converts a time code term into a Unix timestamp in seconds.

Compilation flags:
`static`

Template:
`unix_seconds(TimeCode,Seconds)`
Mode and number of proofs:
`unix_seconds(+compound,-number)` - `one_or_error`

Exceptions:
`TimeCode` is a variable:
`instantiation_error`
`TimeCode` is neither a variable nor a valid time code term for the selected object:
`domain_error(ccsds_time_code_term,TimeCode)`

---

from_unix_seconds/2

Converts a Unix timestamp in seconds into a time code term for the selected object.

Compilation flags:
`static`

Template:
`from_unix_seconds(Seconds,TimeCode)`
Mode and number of proofs:
`from_unix_seconds(+number,-compound)` - `one_or_error`

Exceptions:
`Seconds` is a variable:
`instantiation_error`
`Seconds` cannot be represented using the selected object:
`domain_error(ccsds_time_code_unix_seconds,Seconds)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.37.5 ccsds_time_codes

Helpers for introspecting CCSDS time code terms.

Availability:
`logtalk_load(ccsds_time_codes(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-08

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *valid/1*
  - *format/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

valid/1

True if the argument is a known CCSDS time code term.

Compilation flags:
`static`

Template:
`valid(TimeCode)`
Mode and number of proofs:
`valid(@compound) - zero_or_one`

---

format/2

Returns the CCSDS time code format for a known time code term.

Compilation flags:
`static`

Template:
`format(TimeCode,Format)`
Mode and number of proofs:
`format(+compound,-atom) - one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.37.6 `ccsds_time_codes_types`

Type definitions and arbitrary generators for CCSDS time codes.

Availability:
`logtalk_load(ccsds_time_codes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-25

Compilation flags:
`static`

Provides:
type::type/1
type::check/2
arbitrary::arbitrary/1
arbitrary::arbitrary/2
Uses:
ccsds_ccs(CalendarVariant,FractionOctets)
ccsds_cds(DaySegmentOctets,SubmillisecondOctets,Epoch)
ccsds_cuc(CoarseOctets,FineOctets,Epoch)
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.38 ccsds_time_fields

object

1.38.1 `ccsds_time_fields`

Helpers for parsing and generating self-describing CCSDS time fields using descriptor terms and the existing CCSDS time-code objects.

Availability:
`logtalk_load(ccsds_time_fields(loader))`

Author: Paulo Moura
Version: 1:0:2
Date: 2026-07-20

Compilation flags:
`static, context_switching_calls`

Uses:
reader
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *valid_descriptor/1*
  - *parse/3*
  - *generate/3*

    - *format/2*
    - *epoch/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

valid_descriptor/1

True if the argument is a supported CCSDS time-field descriptor term.

Compilation flags:
`static`

Template:
`valid_descriptor(Descriptor)`
Mode and number of proofs:
`valid_descriptor(@compound)` - `zero_or_one`

---

parse/3

Parses a self-describing CCSDS time field from a source term into a descriptor term and a time-code term. Supported source terms are `file(File)`, `stream(Stream)`, and `bytes(Bytes)`.

Compilation flags:
`static`

Template:
`parse(Source,Descriptor,TimeCode)`
Mode and number of proofs:
`parse(+compound,?compound,-compound)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(ccsds_time_field_source,Source)`
`Source` contents are not a supported CCSDS time field:
`domain_error(ccsds_time_field_byte_sequence,Bytes)`

---

generate/3

Generates a self-describing CCSDS time field to a sink term from a descriptor term and a time-code term. Supported sink terms are `file(File)`, `stream(Stream)`, and `bytes(Bytes)`.

Compilation flags:
`static`

Template:
`generate(Sink,Descriptor,TimeCode)`
Mode and number of proofs:
`generate(+compound,+compound,+compound)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Descriptor` is a variable:
`instantiation_error`
`TimeCode` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid sink:
`domain_error(ccsds_time_field_sink,Sink)`
`Descriptor` is neither a variable nor a supported descriptor term:
`domain_error(ccsds_time_field_descriptor,Descriptor)`
`TimeCode` is neither a variable nor a valid time code term for the selected descriptor:
`domain_error(ccsds_time_code_term,TimeCode)`

---

format/2

Returns the CCSDS time-code format selected by a descriptor term.

Compilation flags:
`static`

Template:
`format(Descriptor,Format)`
Mode and number of proofs:
`format(+compound,-atom)` - `one_or_error`

Exceptions:
`Descriptor` is a variable:
`instantiation_error`
`Descriptor` is neither a variable nor a supported CCSDS time-field descriptor term:
`domain_error(ccsds_time_field_descriptor,Descriptor)`

---

epoch/2

Returns the epoch selected by a descriptor term. Calendar segmented descriptors return `none`.

Compilation flags:
`static`

Template:
`epoch(Descriptor,Epoch)`
Mode and number of proofs:
`epoch(+compound,-atom)` - `one_or_error`

Exceptions:
`Descriptor` is a variable:
`instantiation_error`
`Descriptor` is neither a variable nor a supported CCSDS time-field descriptor term:
`domain_error(ccsds_time_field_descriptor,Descriptor)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.39 character_sets

category

1.39.1 character_set

Shared implementation support category for character set objects.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-08-04

Compilation flags:
`static`

Implements:

`public` *character_set_protocol*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
  - *valid_unicode_scalar/1*
  - *continuation_byte/1*
  - *high_surrogate/1*
  - *low_surrogate/1*
  - *endian_order/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

valid_unicode_scalar/1

True if the argument is a valid Unicode scalar value.

Compilation flags:
`static`

Template:
`valid_unicode_scalar(Code)`
Mode and number of proofs:
`valid_unicode_scalar(+integer)` - `zero_or_one`

---

continuation_byte/1

True if the argument is a valid UTF continuation byte.

Compilation flags:
`static`

Template:
`continuation_byte(Byte)`

Mode and number of proofs:
`continuation_byte(+integer)` - `zero_or_one`

---

high_surrogate/1

True if the argument is a UTF-16 high surrogate code point.

Compilation flags:
`static`

Template:
`high_surrogate(Code)`
Mode and number of proofs:
`high_surrogate(+integer)` - `zero_or_one`

---

low_surrogate/1

True if the argument is a UTF-16 low surrogate code point.

Compilation flags:
`static`

Template:
`low_surrogate(Code)`
Mode and number of proofs:
`low_surrogate(+integer)` - `zero_or_one`

---

endian_order/2

Maps a character set endian parameter to a byte order.

Compilation flags:
`static`

Template:
`endian_order(Endian,Order)`
Mode and number of proofs:
`endian_order(+atom,-atom)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.39.2 `character_set_protocol`

Character set protocol for converting between lists of character codes and lists of bytes.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- Object names: Concrete object names are derived from the preferred IANA MIME names by lowercasing them and replacing hyphens with underscores.
- IANA registry metadata: The metadata predicates reflect the IANA character set registry. When the registry does not define a distinct preferred MIME alias, the `preferred_mime_name/1` and `name/1` predicates return the same atom.
- Unicode encodings: Unicode character sets use Unicode scalar values and do not emit or consume a byte order mark (BOM).

Inherited public predicates:
(none)

- *Public predicates*
  - *preferred_mime_name/1*
  - *name/1*
  - *alias/1*
  - *mibenum/1*
  - *codes_to_bytes/2*
  - *bytes_to_codes/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

preferred_mime_name/1

Preferred MIME name for the character set according to the IANA registry.

Compilation flags:
```
static
```

Template:
```
preferred_mime_name(Name)
```
Mode and number of proofs:
```
preferred_mime_name(?atom) - zero_or_one
```

---

name/1

Registered character set name according to the IANA registry.

Compilation flags:
```
static
```

Template:
```
name(Name)
```
Mode and number of proofs:
```
name(?atom) - zero_or_one
```

---

alias/1

Alias for the character set according to the IANA registry.

Compilation flags:
```
static
```

Template:
```
alias(Alias)
```
Mode and number of proofs:
```
alias(?atom) - zero_or_more
```

---

mibenum/1

MIBenum value for the character set according to the IANA registry.

Compilation flags:
`static`

Template:
`mibenum(MIBenum)`
Mode and number of proofs:
`mibenum(?integer)` - `zero_or_one`

---

codes_to_bytes/2

Converts a list of character codes to the corresponding list of bytes in the character set when all codes are representable.

Compilation flags:
`static`

Template:
`codes_to_bytes(Codes,Bytes)`
Mode and number of proofs:
`codes_to_bytes(+list(integer),--list(byte))` - `zero_or_one`

---

bytes_to_codes/2

Converts a list of bytes in the character set to the corresponding list of character codes when the byte sequence is valid for that character set.

Compilation flags:
`static`

Template:
`bytes_to_codes(Bytes,Codes)`
Mode and number of proofs:
`bytes_to_codes(+list(byte),--list(integer))` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.39.3 `iso_8859_1`

ISO-8859-1 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *single_byte_character_set(255)*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.4 `iso_8859_10`

ISO-8859-10 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.5 `iso_8859_13`

ISO-8859-13 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.6 `iso_8859_14`

ISO-8859-14 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.7 `iso_8859_15`

ISO-8859-15 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.8 `iso_8859_16`

ISO-8859-16 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.9 `iso_8859_2`

ISO-8859-2 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.39.10 `iso_8859_3`

ISO-8859-3 character set encoder and decoder.

Availability:

```
logtalk_load(character_sets(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *mapped_single_byte_character_set*

Remarks:

(none)

Inherited public predicates:

alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.11 `iso_8859_4`

ISO-8859-4 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.12 `iso_8859_9`

ISO-8859-9 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.39.13 `mapped_single_byte_character_set`

Single-byte character set implementation parameterized by sparse byte-to-code mapping facts and undefined bytes.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static`

Extends:
`public` *character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
  - *mapping/2*
  - *undefined/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

mapping/2

Returns, by backtracking, declared byte-to-code mappings.

Compilation flags:
`static`

Template:
`mapping(Byte,Code)`
Mode and number of proofs:
`mapping(?integer,?integer)` - `zero_or_more`

---

undefined/2

Returns, by backtracking, declared undefined byte values.

Compilation flags:
`static`

Template:
`undefined(Byte,Code)`
Mode and number of proofs:
`undefined(?integer,?integer)` - `zero_or_more`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.39.14 single_byte_character_set(MaxCode)

Single-byte character set implementation parameterized by the maximum valid code.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static`

Extends:
`public` *character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.15 `us_ascii`

US-ASCII character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public single_byte_character_set(127)
```

Remarks:

(none)

Inherited public predicates:

alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

category

1.39.16 `utf_16_character_set(Endian)`

UTF-16 character set implementation parameterized by byte order.

Availability:

```
logtalk_load(character_sets(loader))
```

Author: Paulo Moura
Version: 1:1:1
Date: 2026-11-17

Compilation flags:

```
static
```

Extends:

```
public character_set
```

Uses:
byte_order

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.17 `utf_16be`

UTF-16 big-endian character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *utf_16_character_set(big_endian)*

Remarks:

(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.18 `utf_16le`

UTF-16 little-endian character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *utf_16_character_set(little_endian)*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.39.19 `utf_32_character_set(Endian)`

UTF-32 character set implementation parameterized by byte order.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-11-17

Compilation flags:
`static`

Extends:
`public` *character_set*
Uses:
byte_order

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.20 `utf_32be`

UTF-32 big-endian character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *utf_32_character_set(big_endian)*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.21 `utf_32le`

UTF-32 little-endian character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *utf_32_character_set(little_endian)*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.22 `utf_8`

UTF-8 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *character_set*
Extends:
`public` *utf_8_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.23 `utf_8_character_set`

UTF-8 character set implementation.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.24 `windows_1250`

Windows-1250 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.25 `windows_1251`

Windows-1251 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.26 `windows_1252`

Windows-1252 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.27 `windows_1253`

Windows-1253 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.28 `windows_1254`

Windows-1254 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.39.29 `windows_1257`

Windows-1257 character set encoder and decoder.

Availability:
`logtalk_load(character_sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *mapped_single_byte_character_set*

Remarks:
(none)

Inherited public predicates:
alias/1 *bytes_to_codes/2* *codes_to_bytes/2* *mibenum/1* *name/1* *preferred_mime_name/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.40 chi_square_feature_selector

object

1.40.1 `chi_square_feature_selector`

Selects categorical or discretized continuous features using Pearson or Yates chi-square, or uncorrected or bias-corrected Cramer V.

Availability:
`logtalk_load(chi_square_feature_selector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *filter_feature_selector_common*
`public` *feature_discretization*

Uses:
avltree

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_selector/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *feature_scores/2* *learn/2* *learn/3* *option/2* *option/3* *print_selector/1* *selected_features/2* *selector_options/2* *valid_option/1* *valid_options/1* *valid_selector/1*

- *Public predicates*

- *Protected predicates*
- *Private predicates*
  - *score_columns/4*
  - *check_expected_counts/3*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

score_columns/4

Scores prepared columns using sparse contingency counts.

Compilation flags:

```
static
```

Template:

```
score_columns(Columns,Criterion,ExpectedCountPolicy,Scores)
```

Mode and number of proofs:

```
score_columns(+list(compound),+atom,+term,-list(pair)) - one_or_error
```

Exceptions:

A nondegenerate table has an expectation below the requested minimum:

```
domain_error(chi_square_expected_count,Feature-expected(ObservedMinimum,
RequiredMinimum))
```

---

check_expected_counts/3

Checks the requested minimum expectation for nondegenerate tables.

Compilation flags:

```
static
```

Template:

```
check_expected_counts(Policy,Feature,Counts)
```

Mode and number of proofs:

```
check_expected_counts(+term,+atomic,+compound) - one_or_error
```

Exceptions:

A nondegenerate table has an expectation below the requested minimum:

```
domain_error(chi_square_expected_count,Feature-expected(ObservedMinimum,
RequiredMinimum))
```

Operators

(none)

1.41 classification_protocols

category

1.41.1 `classifier_common`

Shared predicates for classifier diagnostics, dataset validation, mixed-feature distance calculations, and export.

Availability:
`logtalk_load(classification_protocols(loader))`

Author: Paulo Moura
Version: 2:0:1
Date: 2026-06-14

Compilation flags:
`static`

Implements:
`public` *classifier_protocol*
Extends:
`public` *options*
Uses:
format
list
population
type

Remarks:
(none)

Inherited public predicates:
check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*

    - *classifier_diagnostics_data/2*
    - *classifier_export_template/4*
    - *classifier_term_template/2*
    - *print_classifier_template/1*
    - *valid_attribute_names/1*
    - *valid_class_values/1*
    - *valid_feature_types/2*
    - *valid_discrete_values/1*
    - *valid_linear_encoders/1*
    - *dataset_attributes/2*
    - *dataset_examples/2*
    - *check_examples_non_empty/2*
    - *check_examples/2*
    - *check_complete_examples/2*
    - *check_complete_examples_nonvar/2*
    - *build_linear_encoders/4*
    - *examples_to_linear_rows/3*
    - *encode_linear_instance/3*
    - *linear_encoders_feature_count/2*
    - *valid_classifier_metadata/2*
    - *valid_classifier_metadata/3*
    - *mixed_feature_distance/5*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

classifier_diagnostics_data/2

Hook predicate that importing classifier implementations must define in order to expose diagnostics metadata.

Compilation flags:

```
static
```

Template:

```
classifier_diagnostics_data(Classifier,Diagnostics)
```

Mode and number of proofs:

```
classifier_diagnostics_data(+compound,-list(compound)) - one
```

classifier_export_template/4

Hook predicate that importing classifier implementations must define in order to expose the exported classifier template for a given functor.

Compilation flags:

```
static
```

Template:

```
classifier_export_template(Dataset,Classifier,Functor,Template)
```

Mode and number of proofs:

```
classifier_export_template(+object_identifier,+compound,+atom,-callable) - one
```

---

classifier_term_template/2

Hook predicate that importing classifier implementations must define in order to expose the learned classifier term template used by pretty-printing helpers.

Compilation flags:

```
static
```

Template:

```
classifier_term_template(Classifier,Template)
```

Mode and number of proofs:

```
classifier_term_template(+compound,-callable) - one
```

---

print_classifier_template/1

Pretty-printing helper predicate used by importing classifier implementations to show the learned classifier term template.

Compilation flags:

```
static
```

Template:

```
print_classifier_template(Classifier)
```

Mode and number of proofs:

```
print_classifier_template(+compound) - one
```

---

valid_attribute_names/1

True when a list of attribute names is a proper list of distinct atoms.

Compilation flags:
`static`

Template:
`valid_attribute_names(AttributeNames)`
Mode and number of proofs:
`valid_attribute_names(+list(atom))` - `zero_or_one`

---

valid_class_values/1

True when a list of class values is a non-empty proper list of distinct atoms.

Compilation flags:
`static`

Template:
`valid_class_values(ClassValues)`
Mode and number of proofs:
`valid_class_values(+list(atom))` - `zero_or_one`

---

valid_feature_types/2

True when a list of feature type tags is non-empty and each tag belongs to the given allowed set.

Compilation flags:
`static`

Template:
`valid_feature_types(FeatureTypes,AllowedTypes)`
Mode and number of proofs:
`valid_feature_types(+list,+list)` - `zero_or_one`

---

valid_discrete_values/1

True when a list of categorical values is non-empty, contains only nonvar terms, and has no duplicates.

Compilation flags:
`static`

Template:
`valid_discrete_values(Values)`
Mode and number of proofs:
`valid_discrete_values(+list)` - `zero_or_one`

---

valid_linear_encoders/1

True when a list of encoders only contains valid `continuous/3` or `categorical/2` encoder terms with distinct attributes.

Compilation flags:
`static`

Template:
`valid_linear_encoders(Encoders)`
Mode and number of proofs:
`valid_linear_encoders(+list(compound))` - `zero_or_one`

---

dataset_attributes/2

Collects the declared dataset attributes and their value domains.

Compilation flags:
`static`

Template:
`dataset_attributes(Dataset,Attributes)`
Mode and number of proofs:
`dataset_attributes(+object_identifier,-list(pair))` - `one`

---

dataset_examples/2

Collects the dataset training examples as `Id-Class-AttributeValues` terms.

Compilation flags:
`static`

Template:
`dataset_examples(Dataset,Examples)`
Mode and number of proofs:
`dataset_examples(+object_identifier,-list(compound))` - `one`

---

check_examples_non_empty/2

Checks that a training example collection is not empty.

Compilation flags:
`static`

Template:
`check_examples_non_empty(Dataset,Examples)`
Mode and number of proofs:
`check_examples_non_empty(+object_identifier,+list)` - `one`

---

check_examples/2

Checks that a training dataset is non-empty, that all example classes belong to the declared class values, and that provided attribute bindings use declared attributes with values matching the declared domains. Missing attribute bindings are allowed.

Compilation flags:
`static`

Template:
`check_examples(Dataset,Examples)`
Mode and number of proofs:
`check_examples(+object_identifier,+list)` - `one_or_error`

Exceptions:
`Examples` is empty:
`domain_error(non_empty_dataset,Dataset)`
`Dataset` class values declaration is invalid:
`domain_error(class_values,ClassValues)`
An example class is not one of the declared class values:
`domain_error(dataset_example_class,Class)`
An example attribute binding uses an undeclared attribute or a value outside the declared attribute domain:

`domain_error(dataset_example_attribute_values,AttributeValues)`

---

check_complete_examples/2

Checks that a training dataset is non-empty, that all example classes belong to the declared class values, and that each example contains every declared attribute exactly once with values matching the declared domains. Missing values represented using variables are allowed.

Compilation flags:

`static`

Template:

`check_complete_examples(Dataset,Examples)`

Mode and number of proofs:

`check_complete_examples(+object_identifier,+list)` - `one_or_error`

Exceptions:

`Examples` is empty:

`domain_error(non_empty_dataset,Dataset)`

`Dataset` class values declaration is invalid:

`domain_error(class_values,ClassValues)`

An example class is not one of the declared class values:

`domain_error(dataset_example_class,Class)`

An example attribute set is missing a declared attribute, contains a duplicate attribute, uses an undeclared attribute, or contains a non-variable value outside the declared attribute domain:

`domain_error(dataset_example_attribute_values,AttributeValues)`

---

check_complete_examples_nonvar/2

Checks that a training dataset is non-empty, that all example classes belong to the declared class values, and that each example contains every declared attribute exactly once with non-variable values matching the declared domains.

Compilation flags:

`static`

Template:

`check_complete_examples_nonvar(Dataset,Examples)`

Mode and number of proofs:

`check_complete_examples_nonvar(+object_identifier,+list)` - `one_or_error`

Exceptions:

`Examples` is empty:

`domain_error(non_empty_dataset,Dataset)`

`Dataset` class values declaration is invalid:

`domain_error(class_values,ClassValues)`

An example class is not one of the declared class values:

```
domain_error(dataset_example_class,Class)
```

An example attribute set is missing a declared attribute, contains a duplicate attribute, uses an undeclared attribute, contains a variable value, or contains a value outside the declared attribute domain:

```
domain_error(dataset_example_attribute_values,AttributeValues)
```

---

build_linear_encoders/4

Builds linear-model encoders for continuous and categorical attributes. Continuous encoders optionally standardize features when `FeatureScaling` is true.

Compilation flags:

```
static
```

Template:

```
build_linear_encoders(Attributes,Examples,FeatureScaling,Encoders)
```

Mode and number of proofs:

```
build_linear_encoders(+list(pair),+list(compound),+boolean,-list(compound)) - one
```

---

examples_to_linear_rows/3

Encodes examples into `Features-Class` rows using the given linear encoders.

Compilation flags:

```
static
```

Template:

```
examples_to_linear_rows(Examples,Encoders,Rows)
```

Mode and number of proofs:

```
examples_to_linear_rows(+list(compound),+list(compound),-list(pair)) - one
```

---

encode_linear_instance/3

Encodes an instance into a numeric feature vector using the given linear encoders.

Compilation flags:

```
static
```

Template:

```
encode_linear_instance(Encoders,Instance,Features)
```

Mode and number of proofs:

```
encode_linear_instance(+list(compound),+list(pair),-list(float)) - one
```

---

linear_encoders_feature_count/2

Computes the number of numeric features produced by a set of linear encoders.

Compilation flags:
```
static
```

Template:
```
linear_encoders_feature_count(Encoders,Count)
```
Mode and number of proofs:
```
linear_encoders_feature_count(+list(compound),-integer) - one
```

---

valid_classifier_metadata/2

True when diagnostics metadata contains the expected model term.

Compilation flags:
```
static
```

Template:
```
valid_classifier_metadata(Model,Diagnostics)
```
Mode and number of proofs:
```
valid_classifier_metadata(+atom,+list(compound)) - zero_or_one
```

---

valid_classifier_metadata/3

True when diagnostics metadata contains the expected model term and records the given effective options.

Compilation flags:
```
static
```

Template:
```
valid_classifier_metadata(Model,Options,Diagnostics)
```
Mode and number of proofs:
```
valid_classifier_metadata(+atom,+list(compound),+list(compound)) - zero_or_one
```

---

mixed_feature_distance/5

Computes a distance between two mixed-feature vectors using the given feature types and one of the supported metrics `euclidean`, `manhattan`, `chebyshev`, `cosine`, or `minkowski(Order)`.

Compilation flags:
`static`

Template:
`mixed_feature_distance(Metric,FeatureTypes,Values1,Values2,Distance)`
Mode and number of proofs:
`mixed_feature_distance(+term,+list,+list,+list,-float)` - `one_or_error`

Exceptions:
`Metric` is `minkowski(Order)` and `Order` is not a positive number:
`domain_error(positive_number,Order)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.41.2 classifier_protocol

Protocol for machine learning classifiers.

Availability:
`logtalk_load(classification_protocols(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2026-08-09

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/3*
  - *learn/2*
  - *predict/3*
  - *check_classifier/1*
  - *valid_classifier/1*
  - *diagnostics/2*
  - *diagnostic/2*
  - *classifier_options/2*
  - *export_to_clauses/4*
  - *export_to_file/4*
  - *print_classifier/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/3

Learns a classifier from the given dataset object using the specified options.

Compilation flags:
```
static
```

Template:
```
learn(Dataset,Classifier,Options)
```
Mode and number of proofs:
```
learn(+object_identifier,-compound,+list(compound)) - one
```

---

learn/2

Learns a classifier from the given dataset object using default options.

Compilation flags:
```
static
```

Template:
```
learn(Dataset,Classifier)
```
Mode and number of proofs:
```
learn(+object_identifier,-compound) - one
```

---

predict/3

Predicts the class label for a new instance using the learned classifier. The instance is a list of `Attribute-Value` pairs.

Compilation flags:
`static`

Template:
`predict(Classifier,Instance,Class)`
Mode and number of proofs:
`predict(+compound,+list,-atom)` - `one`

---

check_classifier/1

Checks that a learned classifier term is structurally valid for the receiving implementation. Throws an exception when the term is not a valid classifier representation.

Compilation flags:
`static`

Template:
`check_classifier(Classifier)`
Mode and number of proofs:
`check_classifier(@compound)` - `one_or_error`

Exceptions:
`Classifier` is insufficiently instantiated for the receiving implementation:
`instantiation_error`
`Classifier` is sufficiently instantiated but is not a valid classifier:
`domain_error(classifier,Classifier)`

---

valid_classifier/1

True when a learned classifier term is structurally valid for the receiving implementation. Succeeds iff `check_classifier/1` succeeds without throwing an exception.

Compilation flags:
`static`

Template:
`valid_classifier(Classifier)`
Mode and number of proofs:
`valid_classifier(@compound)` - `zero_or_one`

---

diagnostics/2

Returns diagnostics and metadata associated with a learned classifier in a representation-independent way.

Compilation flags:
```
static
```

Template:
```
diagnostics(Classifier,Diagnostics)
```
Mode and number of proofs:
```
diagnostics(+compound,-list(compound)) - one
```

---

diagnostic/2

Tests or enumerates individual diagnostics metadata terms for a learned classifier.

Compilation flags:
```
static
```

Template:
```
diagnostic(Classifier,Diagnostic)
```
Mode and number of proofs:
```
diagnostic(+compound,?compound) - zero_or_more
```

---

classifier_options/2

Returns the effective training options recorded in a learned classifier diagnostics list.

Compilation flags:
```
static
```

Template:
```
classifier_options(Classifier,Options)
```
Mode and number of proofs:
```
classifier_options(+compound,-list(compound)) - zero_or_one
```

---

export_to_clauses/4

Converts a classifier into a list of predicate clauses. `Functor` is the functor for the generated predicate clauses. When exporting a serialized classifier term, a noun such as `classifier` or `model` is usually clearer than a verb such as `classify`.

Compilation flags:

```
static
```

Template:

```
export_to_clauses(Dataset,Classifier,Functor,Clauses)
```

Mode and number of proofs:

```
export_to_clauses(+object_identifier,+compound,+callable,-list(clause)) - one
```

---

export_to_file/4

Exports a classifier to a file. `Functor` is the functor for the generated predicate clauses. When exporting a serialized classifier term, a noun such as `classifier` or `model` is usually clearer than a verb such as `classify`.

Compilation flags:

```
static
```

Template:

```
export_to_file(Dataset,Classifier,Functor,File)
```

Mode and number of proofs:

```
export_to_file(+object_identifier,+compound,+callable,+atom) - one
```

---

print_classifier/1

Prints a classifier to the current output stream in a human-readable format.

Compilation flags:

```
static
```

Template:

```
print_classifier(Classifier)
```

Mode and number of proofs:

```
print_classifier(+compound) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

adaptive_boosting_classifier, *c45_classifier*, *gradient_boosting_classifier*, *isolation_forest_anomaly_detector*, *kernel_svm_classifier*, *knn_classifier*, *lda_classifier*, *linear_svm_classifier*, *logistic_regression_classifier*, *mlp_classifier*, *naive_bayes_classifier*, *nearest_centroid_classifier*, *qda_classifier*, *random_forest_classifier*, *sgd_classifier*

protocol

1.41.3 `dataset_protocol`

Protocol for datasets used with classifier algorithms.

Availability:
`logtalk_load(classification_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-19

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - `attribute_values/2`
  - `class/1`
  - `class_values/1`
  - `example/3`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

attribute_values/2

Enumerates by backtracking the attributes and their possible values. For discrete attributes, `Values` is a list of possible values. For continuous (numeric) attributes, `Values` is the atom `continuous`.

Compilation flags:

```
static
```

Template:

```
attribute_values(Attribute,Values)
```

Mode and number of proofs:

```
attribute_values(?atom,-list(atom)) - zero_or_more
attribute_values(?atom,-atom) - zero_or_more
```

---

class/1

Returns the name of the target class attribute.

Compilation flags:

```
static
```

Template:

```
class(Class)
```

Mode and number of proofs:

```
class(-atom) - one
```

---

class_values/1

Returns the list of possible values for the target class attribute.

Compilation flags:

```
static
```

Template:

```
class_values(Values)
```

Mode and number of proofs:

```
class_values(-list(atom)) - one
```

---

example/3

Enumerates by backtracking the examples in the dataset. Each example has an `Id`, a `Class` value, and a list of `Attribute-Value` pairs.

Compilation flags:
`static`

Template:
`example(Id,Class,AttributeValues)`
Mode and number of proofs:
`example(-integer,-atom,-list(pair))` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.41.4 probabilistic_classifier_common

Shared predicates for probabilistic classifiers.

Availability:
`logtalk_load(classification_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-11

Compilation flags:
`static`

Implements:
`public` *probabilistic_classifier_protocol*
Extends:
`public` *classifier_common*

Remarks:
(none)

Inherited public predicates:

check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *predict_from_probabilities/3*
  - *max_probability/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

predict_from_probabilities/3

Predicts the class label for a new instance by selecting the maximum-probability class returned by `predict_probabilities/3`.

Compilation flags:
`static`

Template:
`predict_from_probabilities(Classifier,Instance,Class)`
Mode and number of proofs:
`predict_from_probabilities(+compound,+list,-atom)` - `one`

---

max_probability/3

Selects the class-probability pair with maximum probability from a non-empty list of `Class-Probability` pairs.

Compilation flags:
`static`

Template:
`max_probability(Probabilities,Class,Probability)`
Mode and number of proofs:
`max_probability(+list(pair),-atom,-float)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.41.5 probabilistic_classifier_protocol

Protocol for classifiers that can estimate class probabilities.

Availability:

`logtalk_load(classification_protocols(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-08-09

Compilation flags:

`static`

Extends:

`public` *classifier_protocol*

Remarks:

(none)

Inherited public predicates:

check_classifier/1 *classifier_options/2* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *predict/3* *print_classifier/1* *valid_classifier/1*

- *Public predicates*
  - *predict_probabilities/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

predict_probabilities/3

Predicts class probabilities for a new instance using the learned classifier. The instance is a list of `Attribute-Value` pairs.

Compilation flags:

`static`

Template:

```
predict_probabilities(Classifier,Instance,Probabilities)
```

Mode and number of proofs:

```
predict_probabilities(+compound,+list,-list) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

classifier_protocol, *adaptive_boosting_classifier*, *gradient_boosting_classifier*, *kernel_svm_classifier*, *knn_classifier*, *logistic_regression_classifier*, *mlp_classifier*, *naive_bayes_classifier*, *nearest_centroid_classifier*, *random_forest_classifier*, *sgd_classifier*

1.42 clo_span_pattern_miner

object

1.42.1 `clo_span_pattern_miner`

Closed sequential pattern miner for sequence datasets using closure-aware projected-database search.

Availability:

```
logtalk_load(clo_span_pattern_miner(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public sequential_pattern_mining_common
```

Uses:

format
list

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_pattern_miner/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *mine/2* *mine/3* *option/2* *option/3* *pattern_miner_options/2* *print_pattern_miner/1* *valid_option/1* *valid_options/1* *valid_pattern_miner/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pattern_miner_protocol, *sequence_dataset_protocol*, *prefix_span_pattern_miner*, *gsp_pattern_miner*, *spade_pattern_miner*

1.43 clustering_protocols

category

1.43.1 `clusterer_common`

Shared predicates for clusterer learning defaults, export, and common dataset and encoding helpers.

Availability:

```
logtalk_load(clustering_protocols(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:

```
static
```

Implements:

public *clusterer_protocol*

Extends:

public *options*

Uses:

avltree
format
list
pairs
population
type

Remarks:

(none)

Inherited public predicates:

check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *dataset_attributes/2*
  - *valid_attribute_names/1*
  - *valid_continuous_encoders/1*
  - *valid_discrete_encoders/1*
  - *valid_mixed_encoders/1*
  - *valid_mixed_vectors/2*
  - *valid_clusterer_metadata/3*
  - *valid_diagnostic_count/3*
  - *valid_diagnostic_choice/3*
  - *check_continuous_attributes/1*
  - *check_examples_non_empty/2*
  - *check_examples/3*
  - *check_example_values/2*
  - *check_example_attributes/2*
  - *check_attribute_bindings/2*
  - *attribute_value/3*
  - *build_encoders/4*
  - *known_attribute_values/3*
  - *examples_to_rows/3*
  - *prepare_training_rows/6*
  - *prepare_training_rows/7*
  - *encode_instance/3*
  - *check_encoded_attribute_bindings/2*
  - *normalize_continuous/4*
  - *check_cluster_count/2*
  - *take_first_k/3*

    - *remove_candidate/3*
    - *clusterer_diagnostics_data/2*
  - *Private predicates*
  - *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

dataset_attributes/2

Collects the dataset attribute declarations as *Attribute-Values* pairs.

Compilation flags:
```
static
```

Template:
```
dataset_attributes(Dataset,Attributes)
```
Mode and number of proofs:
```
dataset_attributes(+object_identifier,-list(pair)) - one
```

---

valid_attribute_names/1

True when a list of attribute names is a proper list of distinct atoms.

Compilation flags:
```
static
```

Template:
```
valid_attribute_names(AttributeNames)
```
Mode and number of proofs:
```
valid_attribute_names(+list(atom)) - zero_or_one
```

---

valid_continuous_encoders/1

True when a list of encoders only contains valid continuous encoder terms with distinct attributes.

Compilation flags:
```
static
```

Template:
```
valid_continuous_encoders(Encoders)
```

Mode and number of proofs:

```
valid_continuous_encoders(+list(compound)) - zero_or_one
```

---

valid_discrete_encoders/1

True when a list of encoders only contains valid discrete encoder terms with distinct attributes.

Compilation flags:

```
static
```

Template:

```
valid_discrete_encoders(Encoders)
```

Mode and number of proofs:

```
valid_discrete_encoders(+list(compound)) - zero_or_one
```

---

valid_mixed_encoders/1

True when a list of encoders only contains valid continuous or discrete encoder terms with distinct attributes.

Compilation flags:

```
static
```

Template:

```
valid_mixed_encoders(Encoders)
```

Mode and number of proofs:

```
valid_mixed_encoders(+list(compound)) - zero_or_one
```

---

valid_mixed_vectors/2

True when all vectors conform to the given continuous or discrete encoder specifications.

Compilation flags:

```
static
```

Template:

```
valid_mixed_vectors(Encoders,Vectors)
```

Mode and number of proofs:

```
valid_mixed_vectors(+list(compound),+list) - zero_or_one
```

---

valid_clusterer_metadata/3

True when diagnostics metadata contains the expected model term and records the given effective options.

Compilation flags:
`static`

Template:
`valid_clusterer_metadata(Model,Options,Diagnostics)`
Mode and number of proofs:
`valid_clusterer_metadata(+atom,+list(compound),+list(compound))` - `zero_or_one`

---

valid_diagnostic_count/3

True when diagnostics contains a count term with the given functor and integer value.

Compilation flags:
`static`

Template:
`valid_diagnostic_count(Functor,Diagnostics,Count)`
Mode and number of proofs:
`valid_diagnostic_count(+atom,+list(compound),+integer)` - `zero_or_one`

---

valid_diagnostic_choice/3

True when diagnostics contains a term with the given functor and a value selected from the allowed choices.

Compilation flags:
`static`

Template:
`valid_diagnostic_choice(Functor,Diagnostics,Choices)`
Mode and number of proofs:
`valid_diagnostic_choice(+atom,+list(compound),+list)` - `zero_or_one`

---

check_continuous_attributes/1

Checks that all declared dataset attributes are continuous.

Compilation flags:
```
static
```

Template:
```
check_continuous_attributes(Attributes)
```
Mode and number of proofs:
```
check_continuous_attributes(+list(pair)) - one
```

---

check_examples_non_empty/2

Checks that a training example collection is not empty.

Compilation flags:
```
static
```

Template:
```
check_examples_non_empty(Dataset,Examples)
```
Mode and number of proofs:
```
check_examples_non_empty(+object_identifier,+list) - one
```

---

check_examples/3

Checks that a continuous training dataset is non-empty and that all example values are numeric.

Compilation flags:
```
static
```

Template:
```
check_examples(Dataset,AttributeNames,Examples)
```
Mode and number of proofs:
```
check_examples(+object_identifier,+list(atom),+list) - one
```

---

check_example_values/2

Checks that all example attribute values are present and numeric for the declared attributes.

Compilation flags:
`static`

Template:
`check_example_values(Examples,AttributeNames)`
Mode and number of proofs:
`check_example_values(+list,+list(atom))` - `one`

---

check_example_attributes/2

Checks that a single example contains exactly the declared attributes and that all values are numeric.

Compilation flags:
`static`

Template:
`check_example_attributes(AttributeNames,AttributeValues)`
Mode and number of proofs:
`check_example_attributes(+list(atom),+list(pair))` - `one`

---

check_attribute_bindings/2

Checks that an attribute-value list contains each declared attribute exactly once and no undeclared attributes.

Compilation flags:
`static`

Template:
`check_attribute_bindings(AttributeNames,AttributeValues)`
Mode and number of proofs:
`check_attribute_bindings(+list(atom),+list(pair))` - `one`

---

attribute_value/3

Looks up an attribute value in a list of *Attribute-Value* pairs.

Compilation flags:
```
static
```

Template:
```
attribute_value(Attribute,AttributeValues,Value)
```
Mode and number of proofs:
```
attribute_value(+atom,+list(pair),-term) - one
```

---

build_encoders/4

Builds continuous feature encoders by computing per-attribute centering and optional scaling statistics.

Compilation flags:
```
static
```

Template:
```
build_encoders(AttributeNames,Examples,Options,Encoders)
```
Mode and number of proofs:
```
build_encoders(+list(atom),+list,+list(compound),-list(compound)) - one
```

---

known_attribute_values/3

Collects the known numeric values for a given attribute across the training examples.

Compilation flags:
```
static
```

Template:
```
known_attribute_values(Examples,Attribute,Values)
```
Mode and number of proofs:
```
known_attribute_values(+list,+atom,-list(number)) - one
```

---

examples_to_rows/3

Encodes training examples into *Id-Features* rows using the object-local or imported encoder implementation.

Compilation flags:

```
static
```

Template:

```
examples_to_rows(Examples,Encoders,Rows)
```

Mode and number of proofs:

```
examples_to_rows(+list,+list(compound),-list(pair)) - one
```

---

prepare_training_rows/6

Validates training examples, fits the encoders, and returns encoded rows in dataset order.

Compilation flags:

```
static
```

Template:

```
prepare_training_rows(Dataset,Attributes,Examples,Options,Encoders,Rows)
```

Mode and number of proofs:

```
prepare_training_rows(+object_identifier,+list(pair),+list,+list(compound),
-list(compound),-list(pair)) - one
```

---

prepare_training_rows/7

Prepares training rows for a mixed or discrete-only schema.

Compilation flags:

```
static
```

Template:

```
prepare_training_rows(Kind,Dataset,Attributes,Examples,Options,Encoders,Rows)
```

Mode and number of proofs:

```
prepare_training_rows(+atom,+object_identifier,+list(pair),+list,+list(compound),
-list(compound),-list(pair)) - one
```

---

encode_instance/3

Encodes an instance using the learned continuous attribute encoders after checking that it contains exactly the declared attributes.

Compilation flags:

```
static
```

Template:

```
encode_instance(Encoders,AttributeValues,Features)
```

Mode and number of proofs:

```
encode_instance(+list(compound),+list(pair),-list(number)) - one
```

---

check_encoded_attribute_bindings/2

Checks that an attribute-value list contains each attribute described by the encoders exactly once and no undeclared attributes.

Compilation flags:

```
static
```

Template:

```
check_encoded_attribute_bindings(Encoders,AttributeValues)
```

Mode and number of proofs:

```
check_encoded_attribute_bindings(+list(compound),+list(pair)) - one
```

---

normalize_continuous/4

Normalizes a continuous value using the learned centering and scaling parameters.

Compilation flags:

```
static
```

Template:

```
normalize_continuous(Value,Mean,Scale,Feature)
```

Mode and number of proofs:

```
normalize_continuous(+number,+number,+number,-number) - one
```

---

check_cluster_count/2

Checks that the requested cluster count does not exceed the number of examples.

Compilation flags:
```
static
```

Template:
```
check_cluster_count(K,Count)
```
Mode and number of proofs:
```
check_cluster_count(+integer,+integer) - one
```

---

take_first_k/3

Collects the first *K* vectors from *Id-Vector* rows.

Compilation flags:
```
static
```

Template:
```
take_first_k(K,Rows,Vectors)
```
Mode and number of proofs:
```
take_first_k(+integer,+list,-list) - one
```

---

remove_candidate/3

Removes the first matching *Id-Vector* candidate from a candidate list.

Compilation flags:
```
static
```

Template:
```
remove_candidate(Candidate,Candidates,RemainingCandidates)
```
Mode and number of proofs:
```
remove_candidate(+pair,+list(pair),-list(pair)) - one
```

---

clusterer_diagnostics_data/2

Hook predicate that importing clusterer implementations must define in order to expose diagnostics metadata.

Compilation flags:
```
static
```

Template:
```
clusterer_diagnostics_data(Clusterer,Diagnostics)
```
Mode and number of proofs:
```
clusterer_diagnostics_data(+compound,-list(compound)) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.43.2 clusterer_protocol

Protocol for machine learning clusterers.

Availability:
```
logtalk_load(clustering_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-30

Compilation flags:
```
static
```

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/3*

    - *learn/2*
    - *cluster/3*
    - *diagnostics/2*
    - *check_clusterer/1*
    - *valid_clusterer/1*
    - *diagnostic/2*
    - *clusterer_options/2*
    - *export_to_clauses/4*
    - *export_to_file/4*
    - *print_clusterer/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/3

Learns a clusterer from the given dataset object using the specified options.

Compilation flags:
`static`

Template:
`learn(Dataset,Clusterer,Options)`
Mode and number of proofs:
`learn(+object_identifier,-compound,+list(compound))` - `one`

---

learn/2

Learns a clusterer from the given dataset object using default options.

Compilation flags:
`static`

Template:
`learn(Dataset,Clusterer)`
Mode and number of proofs:
`learn(+object_identifier,-compound)` - `one`

---

cluster/3

Assigns a new instance to a cluster using the learned clusterer. The instance is a list of `Attribute-Value` pairs.

Compilation flags:
`static`

Template:
`cluster(Clusterer,Instance,Cluster)`
Mode and number of proofs:
`cluster(+compound,+list,-ground)` - `one`

---

diagnostics/2

Returns diagnostics and metadata associated with a learned clusterer in a representation-independent way.

Compilation flags:
`static`

Template:
`diagnostics(Clusterer,Diagnostics)`
Mode and number of proofs:
`diagnostics(+compound,-list(compound))` - `one`

---

check_clusterer/1

Checks that a learned clusterer term is structurally valid for the receiving implementation. Throws an exception when the term is not a valid clusterer representation.

Compilation flags:
`static`

Template:
`check_clusterer(Clusterer)`
Mode and number of proofs:
`check_clusterer(@compound)` - `one_or_error`

Exceptions:
`Clusterer` is a variable:
`instantiation_error`
`Clusterer` is neither a variable nor a valid clusterer:
`domain_error(clusterer,Clusterer)`

---

valid_clusterer/1

True when a learned clusterer term is structurally valid for the receiving implementation. Succeeds iff *check_clusterer/1* succeeds without throwing an exception.

Compilation flags:
```
static
```

Template:
```
valid_clusterer(Clusterer)
```
Mode and number of proofs:
```
valid_clusterer(@compound) - zero_or_one
```

---

diagnostic/2

Tests or enumerates individual diagnostics metadata terms for a learned clusterer.

Compilation flags:
```
static
```

Template:
```
diagnostic(Clusterer,Diagnostic)
```
Mode and number of proofs:
```
diagnostic(+compound,?compound) - zero_or_more
```

---

clusterer_options/2

Returns the effective training options recorded in a learned clusterer diagnostics list.

Compilation flags:
```
static
```

Template:
```
clusterer_options(Clusterer,Options)
```
Mode and number of proofs:
```
clusterer_options(+compound,-list(compound)) - one
```

---

export_to_clauses/4

Converts a clusterer into a list of predicate clauses. `Functor` is the functor for the generated predicate clauses.

Compilation flags:
`static`

Template:
`export_to_clauses(Dataset,Clusterer,Functor,Clauses)`
Mode and number of proofs:
`export_to_clauses(+object_identifier,+compound,+callable,-list(clause))` - `one`

---

export_to_file/4

Exports a clusterer to a file. `Functor` is the functor for the generated `Functor(Clusterer)` predicate clause.

Compilation flags:
`static`

Template:
`export_to_file(Dataset,Clusterer,Functor,File)`
Mode and number of proofs:
`export_to_file(+object_identifier,+compound,+callable,+atom)` - `one`

---

print_clusterer/1

Prints a clusterer to the current output stream in a human-readable format.

Compilation flags:
`static`

Template:
`print_clusterer(Clusterer)`
Mode and number of proofs:
`print_clusterer(+compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

clustering_dataset_protocol

protocol

1.43.3 `clustering_dataset_protocol`

Protocol for datasets used with clustering algorithms.

Availability:

```
logtalk_load(clustering_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-23

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *attribute_values/2*
  - *example/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

attribute_values/2

Enumerates by backtracking the attributes and their possible values. For discrete attributes, `Values` is a list of possible values. For continuous (numeric) attributes, `Values` is the atom `continuous`.

Compilation flags:

```
static
```

Template:

```
attribute_values(Attribute,Values)
```

Mode and number of proofs:

```
attribute_values(?atom,-list(atom)) - zero_or_more
attribute_values(?atom,-atom) - zero_or_more
```

---

example/2

Enumerates by backtracking the examples in the dataset. Each example has an `Id` and a list of `Attribute-Value` pairs.

Compilation flags:

```
static
```

Template:

```
example(Id,AttributeValues)
```

Mode and number of proofs:

```
example(-integer,-list(pair)) - zero_or_more
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

category

1.43.4 search_indexing

Shared helpers for adaptive search-index construction and range queries used by clustering libraries.

Availability:
`logtalk_load(clustering_protocols(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-09-12

Compilation flags:
`static`

Uses:
avltree
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *build_auto_search_index/3*
  - *build_metric_tree/3*
  - *build_grid_index/3*
  - *range_query/5*
  - *search_index_cell_size/2*
  - *select_metric_pivot/4*
  - *distance/4*
  - *split_sorted_rows/5*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

build_auto_search_index/3

Builds an adaptive search index by selecting either a grid index or a metric tree based on dataset shape.

Compilation flags:
```
static
```

Template:
```
build_auto_search_index(Rows,Options,SearchIndex)
```
Mode and number of proofs:
```
build_auto_search_index(+list(pair),+list(compound),-compound) - one
```

---

build_metric_tree/3

Builds a metric tree search index for encoded dataset rows.

Compilation flags:
```
static
```

Template:
```
build_metric_tree(Rows,Options,MetricTree)
```
Mode and number of proofs:
```
build_metric_tree(+list(pair),+list(compound),-compound) - one
```

---

build_grid_index/3

Builds a grid-based search index for encoded dataset rows using the object-defined cell size.

Compilation flags:
```
static
```

Template:
```
build_grid_index(Rows,Options,GridIndex)
```
Mode and number of proofs:
```
build_grid_index(+list(pair),+list(compound),-compound) - one
```

---

range_query/5

Queries a search index for rows within the given epsilon distance of an encoded feature vector.

Compilation flags:
```
static
```

Template:
```
range_query(SearchIndex,Vector,Options,Epsilon,Neighbors)
```
Mode and number of proofs:
```
range_query(+compound,+list(number),+list(compound),+number,-list(pair)) - one
```

---

search_index_cell_size/2

Returns the grid cell size to use when constructing a grid-based search index.

Compilation flags:
```
static
```

Template:
```
search_index_cell_size(Options,CellSize)
```
Mode and number of proofs:
```
search_index_cell_size(+list(compound),-number) - one
```

---

select_metric_pivot/4

Selects a pivot row and returns the remaining rows decorated and sorted by distance to that pivot.

Compilation flags:
```
static
```

Template:
```
select_metric_pivot(Rows,Options,Pivot,SortedRows)
```
Mode and number of proofs:
```
select_metric_pivot(+list(pair),+list(compound),-pair,-list(pair)) - one
```

---

distance/4

Computes the distance between two encoded feature vectors using the effective object options.

Compilation flags:
```
static
```

Template:
```
distance(Options,Vector1,Vector2,Distance)
```
Mode and number of proofs:
```
distance(+list(compound),+list(number),+list(number),-number) - one
```

---

split_sorted_rows/5

Splits rows decorated with distances into inner and outer partitions and returns their boundary distances.

Compilation flags:
```
static
```

Template:
```
split_sorted_rows(SortedRows,InnerUpperBound,OuterLowerBound,InnerRows,OuterRows)
```
Mode and number of proofs:
```
split_sorted_rows(+list(pair),-number,-number,-list(pair),-list(pair)) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.44 cmac

object

1.44.1 cmac

CMAC (Cipher-based Message Authentication Code) implementation as specified in NIST SP 800-38B.

Availability:
```
logtalk_load(cmac(loader))
```

Author: Paulo Moura
Version: 1:1:0

Date: 2026-09-25

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *cmac_protocol*

Uses:
list
type

Remarks:
(none)

Inherited public predicates:
digest/4 *digest/5* *hex_digest/4* *hex_digest/5*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

aes128, *aes192*, *aes256*

protocol

1.44.2 cmac_protocol

Protocol for one-shot CMAC digest computation using block cipher objects implementing `block_cipher_prepared_key_protocol`.

Availability:
`logtalk_load(cmac(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *digest/4*
  - *hex_digest/4*
  - *digest/5*
  - *hex_digest/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

digest/4

Computes the full CMAC digest for a block cipher object, key bytes, and message bytes.

Compilation flags:
`static`

Template:
`digest(Cipher,KeyBytes,MessageBytes,DigestBytes)`
Mode and number of proofs:
`digest(+object_identifier,+list(byte),+list(byte),--list(byte))` - `one_or_error`

Exceptions:

`Cipher` or `MessageBytes` is a variable or a partial list:
`instantiation_error`
`Cipher` does not implement the required prepared-key block cipher protocol:
`domain_error(cmac_cipher,Cipher)`
`Cipher` reports a block size other than 8 or 16 bytes:
`domain_error(cmac_block_size,BlockSize)`
`MessageBytes` is neither a variable nor a list:
`type_error(list(byte),MessageBytes)`
`MessageBytes` contains a non-integer byte:
`type_error(integer,Byte)`
`MessageBytes` contains an integer outside the byte range:
`domain_error(byte,Byte)`

---

hex_digest/4

Computes the full CMAC digest for a block cipher object, key bytes, and message bytes, returning a lowercase hexadecimal atom.

Compilation flags:
`static`

Template:
`hex_digest(Cipher,KeyBytes,MessageBytes,HexDigest)`
Mode and number of proofs:
`hex_digest(+object_identifier,+list(byte),+list(byte),--atom)` - `one_or_error`

Exceptions:
`Cipher` or `MessageBytes` is a variable or a partial list:
`instantiation_error`
`Cipher` does not implement the required prepared-key block cipher protocol:
`domain_error(cmac_cipher,Cipher)`
`Cipher` reports a block size other than 8 or 16 bytes:
`domain_error(cmac_block_size,BlockSize)`
`MessageBytes` is neither a variable nor a list:
`type_error(list(byte),MessageBytes)`
An element `Byte` of the *MessageBytes*` list is neither a variable nor an integer:
`type_error(integer,Byte)`
An element `Byte` of the `MessageBytes` list is an integer but not a valid byte:
`domain_error(byte,Byte)`

---

digest/5

Computes a truncated CMAC digest containing the requested number of leftmost bytes.

Compilation flags:
`static`

Template:
`digest(Cipher,KeyBytes,MessageBytes,Length,DigestBytes)`
Mode and number of proofs:
`digest(+object_identifier,+list(byte),+list(byte),+integer,--list(byte))` - `one_or_error`

Exceptions:
`Length` is a variable:
`instantiation_error`
`Length` is neither a variable nor an integer:
`type_error(integer,Length)`
`Length` is an integer outside the range from one through the cipher block size:
`domain_error(cmac_output_length(1,BlockSize),Length)`

---

hex_digest/5

Computes a truncated CMAC digest containing the requested number of leftmost bytes and returns it as a lowercase hexadecimal atom.

Compilation flags:
`static`

Template:
`hex_digest(Cipher,KeyBytes,MessageBytes,Length,HexDigest)`
Mode and number of proofs:
`hex_digest(+object_identifier,+list(byte),+list(byte),+integer,--atom)` - `one_or_error`

Exceptions:
`Cipher`, `MessageBytes`, or `Length` is a variable or a partial list:
`instantiation_error`
`Cipher` does not implement the required prepared-key block cipher protocol:
`domain_error(cmac_cipher,Cipher)`
`Cipher` reports a block size other than 8 or 16 bytes:
`domain_error(cmac_block_size,BlockSize)`
`MessageBytes` is neither a variable nor a list:
`type_error(list(byte),MessageBytes)`
An element `Byte` of the *MessageBytes*` list is neither a variable nor an integer:
`type_error(integer,Byte)`
An element `Byte` of the `MessageBytes` list is an integer but not a valid byte:
`domain_error(byte,Byte)`

`Length` is neither a variable nor an integer:
`type_error(integer,Length)`
`Length` is an integer outside the range from one through the cipher block size:
`domain_error(cmac_output_length(1,BlockSize),Length)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

block_cipher_prepared_key_protocol

1.45 code_metrics

object

1.45.1 `cc_metric`

Cyclomatic complexity metric. All defined predicates that are not called or updated are counted as graph connected components (the reasoning being that these predicates can be considered entry points). The score is represented by a non-negative integer.

Availability:
`logtalk_load(code_metrics(loader))`

Author: Paulo Moura
Version: 0:5:3
Date: 2026-03-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *code_metrics_utilities*
`public` *code_metric*
Provides:
logtalk::message_tokens//2
Uses:
list
logtalk

numberlist

Remarks:
(none)

Inherited public predicates:
all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.45.2 code_metric

Core predicates for computing source code metrics.

Availability:
`logtalk_load(code_metrics(loader))`

Author: Ebrahim Azarisooreh and Paulo Moura
Version: 0:13:0
Date: 2025-10-06

Compilation flags:
`static`

Extends:
`public` *code_metrics_utilities*

`public` *options*

Uses:

list
logtalk
os
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *entity/1*
  - *file/2*
  - *file/1*
  - *directory/2*
  - *directory/1*
  - *rdirectory/2*
  - *rdirectory/1*
  - *library/2*
  - *library/1*
  - *rlibrary/2*
  - *rlibrary/1*
  - *all/1*
  - *all/0*
  - *entity_score/2*
  - *library_score/2*
  - *rlibrary_score/2*
  - *file_score/2*
  - *directory_score/2*
  - *rdirectory_score/2*
  - *all_score/1*
  - *format_entity_score//2*
- *Protected predicates*
  - *process_entity/2*
  - *process_file/2*
  - *process_directory/2*
  - *process_rdirectory/2*
  - *process_library/2*
  - *process_rlibrary/2*
  - *process_all/1*
  - *sub_directory/2*
  - *sub_library/2*
- *Private predicates*
- *Operators*

Public predicates

entity/1

Scans an entity and prints its metric score.

Compilation flags:
`static`

Template:
`entity(Entity)`
Mode and number of proofs:
`entity(+term)` - `zero_or_one`

---

file/2

Prints metric scores for all the entities defined in a loaded source file using the given options.

Compilation flags:
`static`

Template:
`file(File,Options)`
Mode and number of proofs:
`file(+atom,+list(compound))` - `zero_or_one`

---

file/1

Prints metric scores for all the entities defined in a loaded source file using default options.

Compilation flags:
`static`

Template:
`file(File)`
Mode and number of proofs:
`file(+atom)` - `zero_or_one`

---

directory/2

Scans a directory and prints metric scores for all entities defined in its loaded source files using the given options.

Compilation flags:
`static`

Template:
`directory(Directory,Options)`
Mode and number of proofs:
`directory(+atom,+list(compound))` - `one`

---

directory/1

Scans a directory and prints metric scores for all entities defined in its loaded source files using default options.

Compilation flags:
`static`

Template:
`directory(Directory)`
Mode and number of proofs:
`directory(+atom)` - `one`

---

rdirectory/2

Recursive version of the `directory/1` predicate using the given options.

Compilation flags:
`static`

Template:
`rdirectory(Directory,Options)`
Mode and number of proofs:
`rdirectory(+atom,+list(compound))` - `one`

---

rdirectory/1

Recursive version of the `directory/1` predicate using default options.

Compilation flags:
`static`

Template:
`rdirectory(Directory)`
Mode and number of proofs:
`rdirectory(+atom)` - `one`

---

library/2

Prints metrics scores for all loaded entities from a given library using the given options.

Compilation flags:
`static`

Template:
`library(Library,Options)`
Mode and number of proofs:
`library(+atom,+list(compound))` - `one`

---

library/1

Prints metrics scores for all loaded entities from a given library using default options.

Compilation flags:
`static`

Template:
`library(Library)`
Mode and number of proofs:
`library(+atom)` - `one`

---

rlibrary/2

Recursive version of the `library/1` predicate using the given options.

Compilation flags:
`static`

Template:
`rlibrary(Library,Options)`
Mode and number of proofs:
`rlibrary(+atom,+list(compound))` - `one`

---

rlibrary/1

Recursive version of the `library/1` predicate using default options.

Compilation flags:
`static`

Template:
`rlibrary(Library)`
Mode and number of proofs:
`rlibrary(+atom)` - `one`

---

all/1

Scans all loaded entities and prints their metric scores using the given options.

Compilation flags:
`static`

Template:
`all(Options)`
Mode and number of proofs:
`all(+list(compound))` - `one`

---

all/0

Scans all loaded entities and prints their metric scores using default options.

Compilation flags:
`static`

Mode and number of proofs:
`all` - `one`

---

entity_score/2

Score is a term that represents the metric score associated with a loaded entity. Fails if the metric does not apply.

Compilation flags:
`static`

Template:
`entity_score(Entity,Score)`
Mode and number of proofs:
`entity_score(@entity_identifier,-ground)` - `zero_or_one`

---

library_score/2

Score is a term that represents the metric score associated with a loaded library source files. Fails if the metric does not apply.

Compilation flags:
`static`

Template:
`library_score(Library,Score)`
Mode and number of proofs:
`library_score(@atom,-ground)` - `zero_or_one`

---

rlibrary_score/2

Score is a term that represents the metric score associated with loaded source files from a library and its sub-libraries. Fails if the metric does not apply.

Compilation flags:
`static`

Template:

```
rlibrary_score(Library,Score)
```

Mode and number of proofs:

```
rlibrary_score(@atom,-ground) - zero_or_one
```

---

file_score/2

Score is a term that represents the metric score associated with a loaded source file. Fails if the metric does not apply.

Compilation flags:

```
static
```

Template:

```
file_score(File,Score)
```

Mode and number of proofs:

```
file_score(@atom,-ground) - zero_or_one
```

---

directory_score/2

Score is a term that represents the metric score associated with loaded source files from a directory. Fails if the metric does not apply.

Compilation flags:

```
static
```

Template:

```
directory_score(Directory,Score)
```

Mode and number of proofs:

```
directory_score(@atom,-ground) - zero_or_one
```

---

rdirectory_score/2

Score is a term that represents the metric score associated with loaded source files from a directory and its sub-directories. Fails if the metric does not apply.

Compilation flags:

```
static
```

Template:

```
rdirectory_score(Directory,Score)
```

Mode and number of proofs:

```
rdirectory_score(@atom,-ground) - zero_or_one
```

---

all_score/1

Score is a term that represents the metric score associated with all loaded source files. Fails if the metric does not apply.

Compilation flags:
```
static
```

Template:
```
all_score(Score)
```
Mode and number of proofs:
```
all_score(-ground) - zero_or_one
```

---

format_entity_score//2

Formats the entity score for pretty printing.

Compilation flags:
```
static
```

Template:
```
format_entity_score(Entity,Score)
```
Mode and number of proofs:
```
format_entity_score(@entity_identifier,+ground) - one
```

Protected predicates

process_entity/2

Processes an entity of the given kind.

Compilation flags:
```
static
```

Template:
```
process_entity(Kind,Entity)
```
Mode and number of proofs:
```
process_entity(+atom,@entity_identifier) - one
```

---

process_file/2

Processes a source file using the given options.

Compilation flags:
```
static
```

Template:
```
process_file(Path,Options)
```
Mode and number of proofs:
```
process_file(+atom,+list(compound)) - one
```

---

process_directory/2

Processes a directory of source files using the given options.

Compilation flags:
```
static
```

Template:
```
process_directory(Path,Options)
```
Mode and number of proofs:
```
process_directory(+atom,+list(compound)) - one
```

---

process_rdirectory/2

Recursively process a directory of source files using the given options.

Compilation flags:
```
static
```

Template:
```
process_rdirectory(Path,Options)
```
Mode and number of proofs:
```
process_rdirectory(+atom,+list(compound)) - one
```

---

process_library/2

Processes a library of source files using the given options.

Compilation flags:
```
static
```

Template:
```
process_library(Library,Options)
```
Mode and number of proofs:
```
process_library(+atom,+list(compound)) - one
```

---

process_rlibrary/2

Recursively process a library of source files using the given options.

Compilation flags:
```
static
```

Template:
```
process_rlibrary(Library,Options)
```
Mode and number of proofs:
```
process_rlibrary(+atom,+list(compound)) - one
```

---

process_all/1

Processes all loaded source code using the given options.

Compilation flags:
```
static
```

Template:
```
process_all(Options)
```
Mode and number of proofs:
```
process_all(+list(compound)) - one
```

---

sub_directory/2

Enumerates, by backtracking, all directory sub-directories containing loaded files.

Compilation flags:
```
static
```

Template:
```
sub_directory(Directory,SubDirectory)
```
Mode and number of proofs:
```
sub_directory(+atom,-atom) - one
```

---

sub_library/2

Enumerates, by backtracking, all library sub-libraries.

Compilation flags:
```
static
```

Template:
```
sub_library(Library,SubLibrary)
```
Mode and number of proofs:
```
sub_library(+atom,-atom) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.45.3 code_metrics

Helper object to apply all loaded code metrics.

Availability:
```
logtalk_load(code_metrics(loader))
```

Author: Ebrahim Azarisooreh and Paulo Moura
Version: 0:1:0
Date: 2017-12-31

Compilation flags:
```
static, context_switching_calls
```

Imports:
`public` *code_metric*
Uses:
logtalk

Remarks:
(none)

Inherited public predicates:
all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.45.4 `code_metrics_messages`

Message translations for the `code_metrics` tool.

Availability:
`logtalk_load(code_metrics(loader))`

Author: Ebrahim Azarisooreh and Paulo Moura
Version: 0:8:0
Date: 2022-05-05

Compilation flags:
`static`

Provides:
logtalk::message_prefix_stream/4
logtalk::message_tokens//2

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.45.5 `code_metrics_utilities`

Internal predicates for analyzing source code.

Availability:
`logtalk_load(code_metrics(loader))`

Author: Ebrahim Azarisooreh
Version: 0:7:0
Date: 2024-03-28

Compilation flags:
`static`

Uses:
list
logtalk

Remarks:

- Usage: This is meant to be imported by any metric added to the system.
- Predicate Scope: This is meant for internal use by metrics only. As such, all provided predicates are protected.

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
  - *ancestor/4*
  - *current_entity/1*
  - *declares_predicate/2*
  - *defines_predicate/2*
  - *defines_predicate/3*
  - *entity_calls/3*
  - *entity_kind/2*
  - *entity_property/2*
  - *entity_updates/3*
  - *not_excluded_file/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

ancestor/4

True if `Entity` descends from `Ancestor`, and `EntityKind` and `AncestorKind` unify with their respective entity types.

Compilation flags:

```
static
```

Template:

```
ancestor(EntityKind,Entity,AncestorKind,Ancestor)
```

Mode and number of proofs:

```
ancestor(?entity,?entity_identifier,?entity,?entity_identifier) - zero_or_more
```

current_entity/1

True if Entity is a currently loaded entity.

Compilation flags:
static

Template:
current_entity(Entity)
Mode and number of proofs:
current_entity(?entity_identifier) - zero_or_more

---

declares_predicate/2

True if Entity declares Predicate internally.

Compilation flags:
static

Template:
declares_predicate(Entity,Predicate)
Mode and number of proofs:
declares_predicate(?entity_identifier,?predicate_indicator) - zero_or_more

---

defines_predicate/2

True if Entity defines an implementation of Predicate internally. Auxiliary predicates are excluded from results.

Compilation flags:
static

Template:
defines_predicate(Entity,Predicate)
Mode and number of proofs:
defines_predicate(?entity_identifier,?predicate_indicator) - zero_or_more

---

defines_predicate/3

Same as `defines_predicate/2`, except `Property` is unified with a property of the predicate.

Compilation flags:
`static`

Template:
`defines_predicate(Entity,Predicate,Property)`
Mode and number of proofs:
`defines_predicate(?entity_identifier,?predicate_indicator,?term)` - `zero_or_more`

---

entity_calls/3

True if a predicate `Caller` within `Entity` makes a `Call`.

Compilation flags:
`static`

Template:
`entity_calls(Entity,Caller,Call)`
Mode and number of proofs:
`entity_calls(?entity_identifier,?predicate_indicator,?predicate_indicator)` - `zero_or_one`

---

entity_kind/2

True if `Kind` defines `Entity` and is one of category, protocol, or object.

Compilation flags:
`static`

Template:
`entity_kind(Entity,Kind)`
Mode and number of proofs:
`entity_kind(+entity_identifier,-entity)` - `zero_or_one`

---

entity_property/2

True if `Property` is a valid property of `Entity`. Entity can be either a category, a protocol, or an object.

Compilation flags:
`static`

Template:
`entity_property(Entity,Property)`
Mode and number of proofs:
`entity_property(+entity_identifier,-term)` - `zero_or_more`

---

entity_updates/3

True if a predicate `Updater` within `Entity` makes a dynamic update to `Updated` (by using e.g. the `asserta/1` or `retract/1` predicates).

Compilation flags:
`static`

Template:
`entity_updates(Entity,Updater,Updated)`
Mode and number of proofs:
`entity_updates(+entity_identifier,?predicate_indicator,?predicate_indicator)` - `zero_or_one`

---

not_excluded_file/3

True if the file is not being excluded.

Compilation flags:
`static`

Template:
`not_excluded_file(ExcludedFiles,Path,Basename)`
Mode and number of proofs:
`not_excluded_file(+list(atom),+atom,+atom)` - `zero_or_one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.45.6 cogc_metric

Cognitive complexity metric (approximation). For each non-auxiliary predicate, the score contribution is the number of extra clauses (i.e. number of clauses minus one, for multi-clause branching) plus one if the predicate is directly recursive. The entity score is the sum of all predicate contributions. Protocols are not scored as they cannot define predicates.

Availability:

```
logtalk_load(code_metrics(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-14

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public code_metrics_utilities
public code_metric
```

Provides:

logtalk::message_tokens//2

Uses:

list
logtalk
numberlist

Remarks:

- Limitations: The reflection API does not expose control constructs such as cuts, conditionals, or disjunctions. This metric approximates cognitive complexity using only clause-level branching and direct recursion.
- Branching: A predicate with N clauses contributes N-1 to the score, representing the N-1 extra choices a reader must track.
- Recursion: A predicate that directly calls itself (within the same entity) contributes an additional 1 to the score.
- Score interpretation: The score is represented by a non-negative integer. Higher scores indicate predicates or entities that are harder to understand due to more branching choices or self-recursion. A score of 0 means all predicates are single-clause and non-recursive.
- Aggregation: File, directory, and library scores are computed by summing the individual entity scores.

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.45.7 `coupling_metric`

Computes entity efferent coupling, afferent coupling, instability, abstractness, and distance from the main sequence.

Availability:

```
logtalk_load(code_metrics(loader))
```

Author: Ebrahim Azarisooreh and Paulo Moura
Version: 0:15:0
Date: 2026-03-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *code_metrics_utilities*
`public` *code_metric*

Uses:

list

Remarks:

- Efferent coupling (Ce): Number of entities that an entity depends on.
- Afferent coupling (Ca): Number of entities that depend on an entity.
- Instability (I): Computed as `Ce / (Ce + Ca)`. Measures the entity resilience to change. Ranging from 0 to 1, with 0 indicating a maximally stable entity and 1 indicating a maximally unstable entity. Ideally, an entity is either maximally stable or maximally unstable.

- Abstractness (A): Computed as the ratio between the number of static predicates with scope directives without a local definition and the number of static predicates with scope directives. Measures the rigidity of an entity. Ranging from 0 to 1, with 0 indicating a fully concrete entity and 1 indicating a fully abstract entity.
- Distance from main sequence (D): Computed as `abs(A + I - 1)`. Measures how far an entity is from the idealized line `A + I = 1`. Ranging from 0 to 1, with 0 indicating the entity is on the main sequence and 1 indicating it is as far as possible (either maximally abstract and stable, or maximally concrete and unstable).
- Entity score: Represented as the compound term `ce_ca_i_a_d(Ce,Ca,I,A,D)`.
- Dependencies count: Includes direct entity relations plus calls or dynamic updates to predicates in external objects or categories.

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.45.8 `dit_metric`

Analyzes the depth of inheritance for objects, protocols, and categories.

Availability:

```
logtalk_load(code_metrics(loader))
```

Author: Ebrahim Azarisooreh
Version: 0:6:2
Date: 2026-03-13

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *code_metrics_utilities*
`public` *code_metric*

Uses:

numberlist

Remarks:

- Depth: The depth is the maximum length of a node to the root entity. Lower scores are generally better.
- Inheritance: A level of inheritance defined by either one of specialization, instantiation, extension, importation, or implementation.
- Scoring: The maximum path length is determined for each entity in question.

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.45.9 doc_metric

Entity and entity predicates documentation score.

Availability:
`logtalk_load(code_metrics(loader))`

Author: Paulo Moura
Version: 0:13:1
Date: 2026-03-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *code_metrics_utilities*
`public` *code_metric*

Uses:
list
numberlist

Remarks:

- Score range: Score is a integer percentage where a 100% score means that all expected documentation information is present.
- Score weights: The score is split by default between 20% for the entity documentation and 80% for the entity predicates documentation, Can be customized using the predicate `entity_predicates_weights_hook/2`.
- Score customization: The individual scores of entity `info/1` pairs and predicate `info/2` pairs can be customized using the `entity_info_pair_score_hook/3` and `predicate_info_pair_score_hook/4` predicates.

Inherited public predicates:
all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
  - *entity_predicates_weights_hook/2*
  - *entity_info_score_hook/2*
  - *entity_info_pair_score_hook/3*
  - *predicate_mode_score_hook/3*
  - *predicate_mode_score_hook/5*
  - *predicate_info_score_hook/3*
  - *predicate_info_pair_score_hook/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

entity_predicates_weights_hook/2

Relative weight between entity documentation and predicates documentation in percentage. The sum of the two values must be equal to 100.

Compilation flags:

```
dynamic, multifile
```

Template:

```
entity_predicates_weights_hook(EntityWeight,PredicatesWeight)
```

Mode and number of proofs:

```
entity_predicates_weights_hook(?integer,?integer) - zero_or_one
```

---

entity_info_score_hook/2

Maximum score for entity `info/1` directives.

Compilation flags:

```
dynamic, multifile
```

Template:

```
entity_info_score_hook(Entity,MaximumScore)
```

Mode and number of proofs:

```
entity_info_score_hook(?term,?integer) - zero_or_one
```

---

entity_info_pair_score_hook/3

Score for relevant entity `info/1` directive pairs. If defined, the `entity_info_score_hook/2` predicate should be defined accordingly.

Compilation flags:

```
dynamic, multifile
```

Template:

```
entity_info_pair_score_hook(Pair,Entity,Score)
```

Mode and number of proofs:

```
entity_info_pair_score_hook(?callable,?term,?integer) - zero_or_more
```

---

predicate_mode_score_hook/3

Maximum score for predicate `mode/2` directives.

Compilation flags:
`dynamic, multifile`

Template:
`predicate_mode_score_hook(Entity,Predicate,MaximumScore)`
Mode and number of proofs:
`predicate_mode_score_hook(?term,?term,?integer)` - `zero_or_more`

---

predicate_mode_score_hook/5

Score for a predicate `mode/2` directive. If defined, the `predicate_mode_score_hook/3` predicate should be defined accordingly.

Compilation flags:
`dynamic, multifile`

Template:
`predicate_mode_score_hook(Template,Solutions,Entity,Predicate,Score)`
Mode and number of proofs:
`predicate_mode_score_hook(?term,?term,?term,?term,?integer)` - `zero_or_one`

---

predicate_info_score_hook/3

Maximum score for predicate `info/2` directives.

Compilation flags:
`dynamic, multifile`

Template:
`predicate_info_score_hook(Entity,Predicate,MaximumScore)`
Mode and number of proofs:
`predicate_info_score_hook(?term,?term,?integer)` - `zero_or_one`

---

predicate_info_pair_score_hook/4

Score for a predicate `info/2` directive pairs. If defined, the `predicate_info_score_hook/3` predicate should be defined accordingly.

Compilation flags:
`dynamic, multifile`

Template:
`predicate_info_pair_score_hook(Pair,Entity,Predicate,Score)`
Mode and number of proofs:
`predicate_info_pair_score_hook(?callable,?term,?term,?integer)` - `zero_or_more`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.45.10 halstead_metric

Computes Halstead complexity numbers for an entity using a Stroud of 18.

Availability:
`logtalk_load(code_metrics(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2018-06-08

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *halstead_metric(18)*

Remarks:
(none)

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.45.11 `halstead_metric(Stroud)`

- `Stroud` - Coefficient for computing the time required to program.

Computes Halstead complexity numbers for an entity.

Availability:

```
logtalk_load(code_metrics(loader))
```

Author: Paulo Moura
Version: 0:10:0
Date: 2025-01-04

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public code_metrics_utilities
public code_metric
```

Uses:

```
list
numberlist
pairs
```

Remarks:

- Definition of operators: Predicates declared, user-defined, and called are interpreted as operators. Built-in predicates and built-in control constructs are ignored.
- Definition of operands: Predicate arguments are abstracted and interpreted as operands. Note that this definition of operands is a significant deviation from the original definition, which used syntactic literals.
- Pn: Number of distinct predicates (declared, defined, called, or updated).
- PAn: Number of predicate arguments (assumed distinct).
- Cn: Number of predicate calls/updates + number of clauses.
- CAn: Number of predicate call/update arguments + number of clause head arguments.
- EV: Entity vocabulary: `EV = Pn + PAn`.
- EL: Entity length: `EL = Cn + CAn`.
- V: Volume: `V = EL * log2(EV)`.
- D: Difficulty: `D = (Pn/2) * (CAn/An)`.
- E: Effort: `E = D * V`.
- T: Time required to program: `T = E/k` seconds (`k` is the Stroud number; defaults to 18).
- B: Number of delivered bugs: `B = V/3000`.
- Entity score: Represented as the compound term `pn_pan_cn_can_ev_el_v_d_e_t_b/11`.

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.45.12 `lcom_metric`

Lack of Cohesion Of Methods metric (LCOM4).

Availability:
`logtalk_load(code_metrics(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-30

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *code_metrics_utilities*
`public` *code_metric*
Provides:
logtalk::message_tokens//2
Uses:
list
logtalk

Remarks:

- Score computation: Computes the number of connected components in the undirected graph whose nodes are the locally defined predicates and whose edges represent direct internal calls between them.
- Score interpretation: A score of 1 indicates a fully cohesive entity. Higher scores indicate that the entity may benefit from being split. Protocols are not scored as they cannot define predicates.
- Score representation: The score is represented by the compound term `lcom(Components, Predicates)` where `Components` is the number of connected components and `Predicates` is the total number of locally defined (non-auxiliary) predicates.

Inherited public predicates:
all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.45.13 `lines_metric`

Computes the number of lines of code, comments, and blanks by calling cloc and parsing its report file output.

Availability:
`logtalk_load(code_metrics(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-07-22

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *code_metrics_utilities*
`public` *code_metric*
Provides:
logtalk::message_tokens//2
Uses:
atom
list
logtalk
os
user

Remarks:

- Entity line range: Entity scores are computed by querying entity lines(BeginLine, EndLine) and running cloc on a temporary file containing only that line range.
- External dependency: Requires `cloc` to be available in `PATH`.
- Entity score: Represented as the compound term `lines(Code, Comments, Blanks)`.

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.45.14 `mi_metric`

Computes the maintainability index metric based on the Cyclomatic Complexity and the Halstead Volume metrics an entity using a Stroud of 18.

Availability:
`logtalk_load(code_metrics(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-14

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *mi_metric(18)*

Remarks:
(none)

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.45.15 `mi_metric(Stroud)`

- `Stroud` - Halstead metric coefficient for computing the time required to program.

Computes the maintainability index metric based on the Cyclomatic Complexity, Halstead Volume, and lines of code metrics for an entity.

Availability:

```
logtalk_load(code_metrics(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-14

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public code_metrics_utilities
public code_metric
```

Uses:

cc_metric
halstead_metric(Stroud)

lines_metric

Remarks:

- Formula: The original equation is used: `MI = 171 - 5.2 * ln(V) - 0.23 * C - 16.2 * ln(L)`. V is the Halstead volume, C is the Cyclomatic Complexity, and L is the number of code lines.
- Entity score: Represented as the compound term `mi(MI)`.

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.45.16 `noc_metric`

Number of entity clauses metric. The score is represented using the compound term `number_of_clauses(Total, User)`.

Availability:
`logtalk_load(code_metrics(loader))`

Author: Ebrahim Azarisooreh and Paulo Moura
Version: 0:14:2
Date: 2026-03-13

Compilation flags:
`static, context_switching_calls`

Imports:

`public` *code_metrics_utilities*

`public` *code_metric*

Provides:

logtalk::message_tokens//2

Uses:

list

logtalk

Remarks:

(none)

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.45.17 `nor_metric`

Number of entity rules metric. The score is represented using the compound term `number_of_rules(Total, User)`.

Availability:

`logtalk_load(code_metrics(loader))`

Author: Paulo Moura

Version: 0:5:2

Date: 2026-03-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *code_metrics_utilities*
`public` *code_metric*

Provides:
logtalk::message_tokens//2

Uses:
list
logtalk

Remarks:
(none)

Inherited public predicates:
all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.45.18 `rfc_metric`

Response For a Class (RFC) metric. The score is the number of distinct predicates in the response set: the locally defined (non-auxiliary) predicates plus all distinct predicates they directly call or update. Protocols are not scored as they cannot define predicates. The score is represented by a non-negative integer.

Availability:
`logtalk_load(code_metrics(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *code_metrics_utilities*
`public` *code_metric*
Provides:
logtalk::message_tokens//2
Uses:
list
logtalk
numberlist

Remarks:

- Response set: The response set RS(E) for an entity E is the union of its locally defined (non-auxiliary) predicates and all predicates directly called or updated by those predicates.
- Score interpretation: Higher scores indicate entities with more potential execution paths in response to a message, which may increase testing effort.
- Aggregation: When computing scores for files, directories, and libraries, the individual entity scores are summed.

Inherited public predicates:
all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.45.19 `size_metric`

Source code size metric. Returned scores are upper bounds and based solely in source file sizes (expressed in bytes).

Availability:
`logtalk_load(code_metrics(loader))`

Author: Paulo Moura
Version: 0:7:2
Date: 2026-03-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *code_metrics_utilities*
`public` *code_metric*
Provides:
logtalk::message_tokens//2
Uses:
list
logtalk
numberlist
os

Remarks:
(none)

Inherited public predicates:
all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.45.20 `upn_metric`

Number of unique predicates nodes metric. The nodes include called and updated predicates independently of where they are defined. The score is represented by a non-negative integer.

Availability:
```
logtalk_load(code_metrics(loader))
```

Author: Paulo Moura
Version: 0:6:3
Date: 2026-03-13

Compilation flags:
```
static, context_switching_calls
```

Imports:
`public` *code_metrics_utilities*
`public` *code_metric*
Provides:
logtalk::message_tokens//2
Uses:
list
logtalk
numberlist

Remarks:
(none)

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.45.21 `wmc_metric`

Weighted Methods per Class (WMC) metric. Uses unit weights, i.e., the score is the number of locally defined (non-auxiliary) predicates. Protocols are not scored as they cannot define predicates. The score is represented by a non-negative integer.

Availability:

```
logtalk_load(code_metrics(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public code_metrics_utilities
public code_metric
```

Provides:

logtalk::message_tokens//2

Uses:

list
logtalk
numberlist

Remarks:

- Unit weights: Each predicate is assigned a weight of 1. A cyclomatic-complexity-weighted variant is not provided as the Logtalk reflection API does not expose intra-clause branching structure (if-then-else, disjunction), which would produce the same approximations as the `cc_metric`.
- Interpretation: Higher scores indicate an entity with more responsibilities that may benefit from being split.
- Aggregation: When computing scores for files, directories, and libraries, the individual entity scores are summed.

Inherited public predicates:

all/0 all/1 all_score/1 check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 directory_score/2 entity/1 entity_score/2 file/1 file/2 file_score/2 format_entity_score//2 library/1 library/2 library_score/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory_score/2 rlibrary/1 rlibrary/2 rlibrary_score/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.46 colley_ranker

object

1.46.1 colley_ranker

Colley pairwise preference ranker. Learns one deterministic rating per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol by solving the Colley linear system built from aggregated pairwise outcomes and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:

```
logtalk_load(colley_ranker(loader))
```

Author: Paulo Moura
Version: 1:2:0
Date: 2026-09-24

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *ranking_dataset_common*
`public` *score_ranker_model_common*
`public` *pairwise_strength_ranker_common*

Uses:

avltree
list
numberlist

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*

- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *copeland_ranker*, *rank_centrality*

1.47 combinations

object

1.47.1 `combinations`

Implementation of combinations operations over lists.

Availability:
`logtalk_load(combinations(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2026-05-21

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *combinations_protocol*
Uses:
fast_random(Algorithm)
list
natural

Remarks:
(none)

Inherited public predicates:

combination/3 *combination/4* *combination_index/4* *combination_index/5* *combinations/3* *combinations/4* *count_combinations/3* *count_distinct_combinations/3* *distinct_combination/3* *distinct_combination/4* *distinct_combination_index/4* *distinct_combinations/3* *distinct_combinations/4* *next_combination/3* *nth_combination/4* *nth_combination/5* *nth_distinct_combination/4* *previous_combination/3* *random_combination/3* *random_distinct_combination/3* *sample_combinations/4* *sample_distinct_combinations/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.47.2 `combinations_protocol`

Protocol for combinations operations over lists.

Availability:

```
logtalk_load(combinations(loader))
```

Author: Paulo Moura
Version: 2:0:0
Date: 2026-05-12

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *combinations/3*
  - *combination/3*
  - *combinations/4*
  - *combination/4*
  - *distinct_combinations/3*
  - *distinct_combination/3*
  - *distinct_combinations/4*
  - *distinct_combination/4*
  - *nth_combination/4*
  - *nth_combination/5*
  - *combination_index/4*
  - *combination_index/5*
  - *count_combinations/3*
  - *count_distinct_combinations/3*
  - *nth_distinct_combination/4*
  - *distinct_combination_index/4*
  - *random_combination/3*
  - *sample_combinations/4*
  - *random_distinct_combination/3*
  - *sample_distinct_combinations/4*
  - *next_combination/3*
  - *previous_combination/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

combinations/3

Generates all K-element combinations of a list using default order.

Compilation flags:

```
static
```

Template:

```
combinations(K,List,Combinations)
```

Mode and number of proofs:

```
combinations(+integer,+list,-list) - one
```

combination/3

True iff the third argument is a K-element combination of a list using default order.

Compilation flags:
`static`

Template:
`combination(K,List,Combination)`
Mode and number of proofs:
`combination(+integer,+list,-list)` - `one_or_more`

---

combinations/4

Generates all K-element combinations with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`combinations(K,List,Order,Combinations)`
Mode and number of proofs:
`combinations(+integer,+list,+atom,-list)` - `one`

---

combination/4

True iff the fourth argument is a K-element combination with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`combination(K,List,Order,Combination)`
Mode and number of proofs:
`combination(+integer,+list,+atom,-list)` - `one_or_more`

---

distinct_combinations/3

Generates all distinct K-element combinations of a list (deduplicating equal-valued combinations) using default order.

Compilation flags:
`static`

Template:
`distinct_combinations(K,List,Combinations)`
Mode and number of proofs:
`distinct_combinations(+integer,+list,-list)` - `one`

---

distinct_combination/3

True iff the third argument is a distinct K-element combination of a list using default order.

Compilation flags:
`static`

Template:
`distinct_combination(K,List,Combination)`
Mode and number of proofs:
`distinct_combination(+integer,+list,-list)` - `one_or_more`

---

distinct_combinations/4

Generates all distinct K-element combinations with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`distinct_combinations(K,List,Order,Combinations)`
Mode and number of proofs:
`distinct_combinations(+integer,+list,+atom,-list)` - `one`

---

distinct_combination/4

True iff the fourth argument is a distinct K-element combination with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`distinct_combination(K,List,Order,Combination)`
Mode and number of proofs:
`distinct_combination(+integer,+list,+atom,-list)` - `one_or_more`

---

nth_combination/4

Returns the combination at a given zero-based index using default order.

Compilation flags:
`static`

Template:
`nth_combination(K,List,Index,Combination)`
Mode and number of proofs:
`nth_combination(+integer,+list,+integer,-list)` - `zero_or_one`

---

nth_combination/5

Returns the combination at a given zero-based index in the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`nth_combination(K,List,Order,Index,Combination)`
Mode and number of proofs:
`nth_combination(+integer,+list,+atom,+integer,-list)` - `zero_or_one`

---

combination_index/4

Returns the zero-based index of a combination using default order.

Compilation flags:
`static`

Template:
`combination_index(K,List,Combination,Index)`
Mode and number of proofs:
`combination_index(+integer,+list,+list,-integer)` - `zero_or_one`

---

combination_index/5

Returns the zero-based index of a combination in the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`combination_index(K,List,Order,Combination,Index)`
Mode and number of proofs:
`combination_index(+integer,+list,+atom,+list,-integer)` - `zero_or_one`

---

count_combinations/3

Counts the number of K-element combinations of a list.

Compilation flags:
`static`

Template:
`count_combinations(K,List,Count)`
Mode and number of proofs:
`count_combinations(+integer,+list,-integer)` - `one`

---

count_distinct_combinations/3

Counts the number of distinct K-element combinations of a list (deduplicating equal-valued combinations).

Compilation flags:
`static`

Template:
`count_distinct_combinations(K,List,Count)`
Mode and number of proofs:
`count_distinct_combinations(+integer,+list,-integer) - one`

---

nth_distinct_combination/4

Returns the distinct combination at a given zero-based index in default generation order.

Compilation flags:
`static`

Template:
`nth_distinct_combination(K,List,Index,Combination)`
Mode and number of proofs:
`nth_distinct_combination(+integer,+list,+integer,-list) - zero_or_one`

---

distinct_combination_index/4

Returns the zero-based index of a distinct combination in default generation order.

Compilation flags:
`static`

Template:
`distinct_combination_index(K,List,Combination,Index)`
Mode and number of proofs:
`distinct_combination_index(+integer,+list,+list,-integer) - zero_or_one`

---

random_combination/3

Returns a random K-element combination of a list.

Compilation flags:
`static`

Template:
`random_combination(K,List,Combination)`
Mode and number of proofs:
`random_combination(+integer,+list,-list)` - `zero_or_one`

---

sample_combinations/4

Returns SampleCount random K-element combinations of a list, sampled with replacement.

Compilation flags:
`static`

Template:
`sample_combinations(K,List,SampleCount,Samples)`
Mode and number of proofs:
`sample_combinations(+integer,+list,+integer,-list)` - `zero_or_one`

---

random_distinct_combination/3

Returns a random distinct K-element combination of a list (deduplicating equal-valued combinations).

Compilation flags:
`static`

Template:
`random_distinct_combination(K,List,Combination)`
Mode and number of proofs:
`random_distinct_combination(+integer,+list,-list)` - `zero_or_one`

---

sample_distinct_combinations/4

Returns SampleCount random distinct K-element combinations of a list, sampled with replacement after deduplicating equal-valued combinations.

Compilation flags:

```
static
```

Template:

```
sample_distinct_combinations(K,List,SampleCount,Samples)
```

Mode and number of proofs:

```
sample_distinct_combinations(+integer,+list,+integer,-list) - zero_or_one
```

---

next_combination/3

Returns the next distinct combination value in lexicographic order induced by the first argument.

Compilation flags:

```
static
```

Template:

```
next_combination(List,Combination,Next)
```

Mode and number of proofs:

```
next_combination(+list,+list,-list) - zero_or_one
```

---

previous_combination/3

Returns the previous distinct combination value in lexicographic order induced by the first argument.

Compilation flags:

```
static
```

Template:

```
previous_combination(List,Combination,Previous)
```

Mode and number of proofs:

```
previous_combination(+list,+list,-list) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.48 command_line_options

category

1.48.1 `command_line_option`

Category for defining command-line options. Import this category into objects that represent individual command-line options and override the predicates as needed.

Availability:
`logtalk_load(command_line_options(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-13

Compilation flags:
`static`

Uses:
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *check/0*
  - *valid/0*
  - *name/1*
  - *short_flags/1*
  - *long_flags/1*
  - *type/1*
  - *default/1*

    - *meta/1*
    - *help/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

check/0

Checks if the command-line option definition is valid. Throws an error if the definition is invalid.

Compilation flags:

`static`

Mode and number of proofs:

`check` - `one_or_error`

Exceptions:

The importing object does not define the `name/1` predicate:

`existence_error(procedure,name/1)`

The argument of the `name/1` predicate is a variable:

`instantiation_error`

The argument `Name` of the `name/1` predicate is neither a variable nor an atom:

`type_error(atom,Name)`

The importing returns values with invalid types for the option declaration predicates:

`error`

---

valid/0

Succeeds if the command-line option definition is valid. Fails otherwise.

Compilation flags:

`static`

Mode and number of proofs:

`valid` - `zero_or_one`

---

name/1

Name used to identify this option in the parsed results. This predicate must be overridden. No default.

Compilation flags:
`static`

Template:
`name(Name)`
Mode and number of proofs:
`name(?atom)` - `zero_or_one`

---

short_flags/1

List of single-character short flags for this option (e.g., `[v]` for `-v`). Default is an empty list.

Compilation flags:
`static`

Template:
`short_flags(Flags)`
Mode and number of proofs:
`short_flags(-list(character))` - `one`

---

long_flags/1

List of long flags for this option (e.g., `[verbose]` for `--verbose`). Default is an empty list.

Compilation flags:
`static`

Template:
`long_flags(Flags)`
Mode and number of proofs:
`long_flags(-list(atom))` - `one`

---

type/1

Option value type. One of `boolean`, `atom`, `integer`, `float`, or `term`. Default is `term`.

Compilation flags:
`static`

Template:
`type(Type)`
Mode and number of proofs:
`type(-atom)` - `one`

---

default/1

Default value for this option if any.

Compilation flags:
`static`

Template:
`default(Default)`
Mode and number of proofs:
`default(-term)` - `zero_or_one`

---

meta/1

Metasyntactic variable name for the help text (e.g., `'FILE'`). Default is an empty atom.

Compilation flags:
`static`

Template:
`meta(Meta)`
Mode and number of proofs:
`meta(-atom)` - `one`

---

help/1

Help text for this option. Can be an atom or a list of atoms for pre-broken lines. Default is an empty atom.

Compilation flags:
`static`

Template:
`help(Help)`
Mode and number of proofs:
`help(-atom)` - `one`
`help(-list(atom))` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

command_line_options

object

1.48.2 command_line_options

Command line options parsing predicates. Uses object-based option specifications with the `command_line_option` category.

Availability:
`logtalk_load(command_line_options(loader))`

Author: Marcus Uneson and Paulo Moura
Version: 1:1:1
Date: 2026-10-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
atom

list
meta
numberlist
term_io
type
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *parse/4*
  - *parse/5*
  - *help/2*
  - *help/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/4

Parses the arguments `AppArguments` according to the option objects `OptionObjects` using default parsing options.

Compilation flags:
`static`

Template:
`parse(OptionObjects,AppArguments,Options,PositionalArguments)`
Mode and number of proofs:
`parse(+list(object),+list(atom),-list,-list(atom))` - `one_or_error`

Exceptions:
`OptionObjects` is a variable, a partial list, or contains an option object definition with an insufficiently instantiated value:
`instantiation_error`
`OptionObjects` is neither a variable nor a list of valid object identifiers:
`type_error(list(object),OptionObjects)`
An option object is missing the `name/1` predicate:
`existence_error(predicate,name/1)`
An option object predicate returns a value with an invalid type:
`type_error(Type,Term)`
An option object predicate returns a value outside the expected domain:
`domain_error(Domain,Term)`
The option set contains a duplicate option name:

`domain_error(unique_key,Name)`
The option set contains a duplicate short flag:
`domain_error(unique_short_flag,ShortFlag)`
The option set contains a duplicate long flag:
`domain_error(unique_long_flag,LongFlag)`
`AppArguments` is a variable or a partial list:
`instantiation_error`
`AppArguments` is neither a variable nor a list of atoms:
`type_error(list(atom),AppArguments)`
An unknown flag is found:
`existence_error(command_line_option,Argument)`
A flag value cannot be parsed as the option type:
`type_error(flag_value,Type)`
A short option is written using the disallowed `-f=value` syntax:
`syntax_error(disallowed: <shortflag>=<value>)`

---

parse/5

Parses the arguments `AppArguments` according to the option objects `OptionObjects` and the parsing options `ParseOptions`. `Options` is a list of parsed options as `Name(Value)` terms by default (or `Func(Name,Value)` when the `output_functor(Func)` parse option is used). `PositionalArguments` are the remaining non-dashed arguments. `ParseOptions` include `output_functor(Func)`, `duplicated_flags(Keep)` (one of `keepfirst`, `keeplast`, `keepall`; default `keeplast`), and `allow_empty_flag_spec(Bool)` (default `true`).

Compilation flags:
`static`

Template:
`parse(OptionObjects,AppArguments,Options,PositionalArguments,ParseOptions)`
Mode and number of proofs:
`parse(+list(object),+list(atom),-list,-list(atom),+list)` - `one_or_error`

Exceptions:
`ParseOptions` is a variable, a partial list, or contains a variable option:
`instantiation_error`
`ParseOptions` is neither a variable nor a list:
`type_error(list,ParseOptions)`
An element of `ParseOptions` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element of `ParseOptions` is a compound term but not a valid option:
`domain_error(option,Option)`
`OptionObjects` is a variable, a partial list, or contains an option object definition with an insufficiently instantiated value:
`instantiation_error`
`OptionObjects` is neither a variable nor a list of valid object identifiers:
`type_error(list(object),OptionObjects)`
An option object is missing the `name/1` predicate:
`existence_error(predicate,name/1)`
An option object predicate returns a value with an invalid type:

`type_error(Type,Term)`

An option object predicate returns a value outside the expected domain:

`domain_error(Domain,Term)`

The option set contains a duplicate option name:

`domain_error(unique_key,Name)`

The option set contains a duplicate short flag:

`domain_error(unique_short_flag,ShortFlag)`

The option set contains a duplicate long flag:

`domain_error(unique_long_flag,LongFlag)`

An option has no short or long flags and empty flag specifications are disallowed:

`domain_error(non_empty_flag_spec,Name)`

`AppArguments` is a variable or a partial list:

`instantiation_error`

`AppArguments` is neither a variable nor a list of atoms:

`type_error(list(atom),AppArguments)`

An unknown flag is found:

`existence_error(command_line_option,Argument)`

A flag value cannot be parsed as the option type:

`type_error(flag_value,Type)`

A short option is written using the disallowed `-f=value` syntax:

`syntax_error(disallowed: <shortflag>=<value>)`

---

help/2

Synthesizes a help text `Help` as an atom from the option objects `OptionObjects` using default help options.

Compilation flags:

`static`

Template:

`help(OptionObjects,Help)`

Mode and number of proofs:

`help(+list(object),-atom)` - `one_or_error`

Exceptions:

`OptionObjects` is a variable, a partial list, or contains an option object definition with an insufficiently instantiated value:

`instantiation_error`

`OptionObjects` is neither a variable nor a list of valid object identifiers:

`type_error(list(object),OptionObjects)`

An option object is missing the `name/1` predicate:

`existence_error(predicate,name/1)`

An option object predicate returns a value with an invalid type:

`type_error(Type,Term)`

An option object predicate returns a value outside the expected domain:

`domain_error(Domain,Term)`

The option set contains a duplicate option name:

`domain_error(unique_key,Name)`

The option set contains a duplicate short flag:
`domain_error(unique_short_flag,ShortFlag)`
The option set contains a duplicate long flag:
`domain_error(unique_long_flag,LongFlag)`

---

help/3

Synthesizes a help text `Help` as an atom from the option objects `OptionObjects` using the given `HelpOptions`. `HelpOptions` include `line_width(Width)` (default 80), `min_help_width(Width)` (default 40), `break_long_flags(Boolean)` (default `false`), `suppress_empty_meta(Boolean)` (default `true`), and `allow_empty_flag_spec(Bool)` (default `true`).

Compilation flags:
`static`

Template:
`help(OptionObjects,Help,HelpOptions)`
Mode and number of proofs:
`help(+list(object),-atom,+list)` - `one_or_error`

Exceptions:
`HelpOptions` is a variable, a partial list, or contains a variable option:
`instantiation_error`
`HelpOptions` is neither a variable nor a list:
`type_error(list,HelpOptions)`
An element of `HelpOptions` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element of `HelpOptions` is a compound term but not a valid option:
`domain_error(option,Option)`
`OptionObjects` is a variable, a partial list, or contains an option object definition with an insufficiently instantiated value:
`instantiation_error`
`OptionObjects` is neither a variable nor a list of valid object identifiers:
`type_error(list(object),OptionObjects)`
An option object is missing the `name/1` predicate:
`existence_error(predicate,name/1)`
An option object predicate returns a value with an invalid type:
`type_error(Type,Term)`
An option object predicate returns a value outside the expected domain:
`domain_error(Domain,Term)`
The option set contains a duplicate option name:
`domain_error(unique_key,Name)`
The option set contains a duplicate short flag:
`domain_error(unique_short_flag,ShortFlag)`
The option set contains a duplicate long flag:
`domain_error(unique_long_flag,LongFlag)`
An option has no short or long flags and empty flag specifications are disallowed:
`domain_error(non_empty_flag_spec,Name)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

command_line_option

1.49 constrained_optimization

object

1.49.1 `augmented_lagrangian(Problem,InnerSolver)`

- `Problem` - Problem object implementing `constrained_optimization_problem_protocol`.
- `InnerSolver` - Atom naming a local_optimization solver class taking a single problem parameter, e.g. `bfgs` or `lbfgs`. Constructed dynamically each outer iteration as `InnerSolver(SubProblem)`.

Augmented Lagrangian method for problems with general equality and inequality constraints, delegating each penalized subproblem to an existing local optimization solver.

Availability:
`logtalk_load(constrained_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *constrained_optimization_solver*
`public` *local_optimization_solver(Problem)*

Uses:
linear_algebra
list

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

constrained_optimization_problem_protocol, *sqp_active_set(Problem)*, *local_optimization_solver(Problem)*, *bfgs(Problem)*, *lbfgs(Problem)*

object

1.49.2 barrier_sub_problem(Problem,OuterIteration,StartPoint,Mu,Rho,Sign)

- `Problem` - The original constrained problem.
- `OuterIteration` - One-based outer iteration number.
- `StartPoint` - Point the inner solve should start from (the current outer iterate, always strictly feasible with respect to the inequality constraints).
- `Mu` - Current barrier weight.
- `Rho` - Current equality-penalty weight.
- `Sign` - 1.0 to minimize the original objective, -1.0 to maximize it.

Internal log-barrier subproblem: objective(x) = Sign*f(x) - mu*sum_j log(-h_j(x)) + (rho/2)*||g(x)||^2, with the barrier term replaced, per inequality constraint, by a smooth large-coefficient quadratic penalty whenever that constraint is not comfortably strictly satisfied. Instantiated fresh by log_barrier(_,_) every outer iteration, starting from StartPoint (the previous outer iterate); not meant to be used directly.

Availability:

`logtalk_load(constrained_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *local_optimization_problem_protocol*
Imports:
`public` *constrained_optimization_solver*
Uses:
linear_algebra
list

Remarks:
(none)

Inherited public predicates:
gradient/2 *hessian/2* *initial_point/1* *objective/2* *position_bounds/1* *progress/5* *stop_condition/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.49.3 constrained_optimization_problem_protocol

Extends `local_optimization_problem_protocol` with general equality and inequality constraints. Box constraints continue to be expressed via the inherited `position_bounds/1`. A problem that defines neither `equality_constraints/2` nor `inequality_constraints/2` is a plain `local_optimization_problem_protocol` problem and can still be solved by any `constrained_optimization` solver.

Availability:

```
logtalk_load(constrained_optimization(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:

```
static
```

Extends:

`public` *local_optimization_problem_protocol*

Remarks:

- Equality constraints: `g(x) = 0`, one component per row of `equality_constraints/2`.
- Inequality constraints: `h(x) =< 0`, one component per row of `inequality_constraints/2`. Constraints of the form `h(x) >= 0` or `a =< h(x) =< b` must be restated in this form by the problem (e.g. negate, or split into two rows) before being reported here.
- Jacobians: `equality_jacobian/2` and `inequality_jacobian/2` are required by `sqp_active_set(_)` and `primal_dual_interior_point(_)`, and by `augmented_lagrangian(_,_)`, `quadratic_penalty(_,_)`, and `log_barrier(_,_)` when their selected inner solver uses gradients. Solvers that need a Jacobian a problem does not define raise an existence error rather than silently falling back to finite differences.

Inherited public predicates:

gradient/2 *hessian/2* *initial_point/1* *objective/2* *position_bounds/1* *progress/5* *stop_condition/3*

- *Public predicates*
  - *equality_constraints/2*
  - *equality_jacobian/2*
  - *inequality_constraints/2*
  - *inequality_jacobian/2*
  - *inner_progress/6*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

equality_constraints/2

Computes `g(x)`, the vector of equality constraint values at a point; feasibility requires every component to equal zero. Optional: when not defined, the problem has no equality constraints.

Compilation flags:

```
static
```

Template:

```
equality_constraints(Point,Values)
```

Mode and number of proofs:

```
equality_constraints(+list(number),-list(number)) - zero_or_one
```

---

equality_jacobian/2

Computes the Jacobian of `equality_constraints/2` at a point, one row per constraint, one column per variable. Optional unless required by the solver in use (see the "Jacobians" remark above).

Compilation flags:

```
static
```

Template:

```
equality_jacobian(Point,Jacobian)
```

Mode and number of proofs:

```
equality_jacobian(+list(number),-list(list(number))) - zero_or_one
```

---

inequality_constraints/2

Computes `h(x)`, the vector of inequality constraint values at a point; feasibility requires every component to be `=< 0`. Optional: when not defined, the problem has no general inequality constraints (position_bounds/1 may still apply).

Compilation flags:

```
static
```

Template:

```
inequality_constraints(Point,Values)
```

Mode and number of proofs:

```
inequality_constraints(+list(number),-list(number)) - zero_or_one
```

---

inequality_jacobian/2

Computes the Jacobian of `inequality_constraints/2` at a point. Optional unless required by the solver in use (see the "Jacobians" remark above).

Compilation flags:
`static`

Template:
`inequality_jacobian(Point,Jacobian)`
Mode and number of proofs:
`inequality_jacobian(+list(number),-list(list(number)))` - `zero_or_one`

---

inner_progress/6

Optional callback reporting progress from an inner solver used by a delegated constrained solver. The stage is `outer(N)` for an outer iteration or `phase1` for a feasibility search. The value and measure are those of the transformed inner subproblem.

Compilation flags:
`static`

Template:
`inner_progress(Stage,Iteration,Point,Value,Measure,Evaluations)`
Mode and number of proofs:
`inner_progress(+term,+non_negative_integer,+list(number),+number,+number,`
`+non_negative_integer)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

local_optimization_problem_protocol, *qp_active_set*, *sqp_active_set(Problem)*, *augmented_lagrangian(Problem,InnerSolver)*, *quadratic_penalty(Problem,InnerSolver)*, *log_barrier(Problem,InnerSolver)*, *primal_dual_interior_point(Problem)*

category

1.49.4 constrained_optimization_solver

Shared numeric, objective-direction, and validation predicates for constrained optimization solvers and their internal problem wrappers.

Availability:
`logtalk_load(constrained_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *abs_max/2*
  - *positive_max/2*
  - *direction_sign/2*
  - *objective_direction/2*
  - *build_inner/3*
  - *update_penalty/6*
  - *abs_sum/2*
  - *validate_numeric_vector/2*
  - *validate_constraint_data/4*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

abs_max/2

Infinity norm: max_i abs(Vec_i), 0.0 for an empty vector.

Compilation flags:
`static`

Template:
`abs_max(Vec,Max)`
Mode and number of proofs:
`abs_max(+list(number),-number)` - `one`

---

positive_max/2

Infinity norm of the positive part: max_i max(0, Vec_i), 0.0 for an empty vector.

Compilation flags:
`static`

Template:
`positive_max(Vec,Max)`
Mode and number of proofs:
`positive_max(+list(number),-number)` - `one`

---

direction_sign/2

Converts an objective direction into its numeric sign.

Compilation flags:
`static`

Template:
`direction_sign(Direction,Sign)`
Mode and number of proofs:
`direction_sign(+atom,-number)` - `one`

---

objective_direction/2

Converts an objective numeric sign into its direction.

Compilation flags:
```
static
```

Template:
```
objective_direction(Sign,Direction)
```
Mode and number of proofs:
```
objective_direction(+number,-atom) - one
```

---

build_inner/3

Constructs a parametric inner solver object from its class atom and problem object.

Compilation flags:
```
static
```

Template:
```
build_inner(InnerSolver,Problem,Solver)
```
Mode and number of proofs:
```
build_inner(+atom,+nonvar,-nonvar) - one
```

---

update_penalty/6

Scales a penalty when the constraint violation does not decrease by the required factor.

Compilation flags:
```
static
```

Template:
```
update_penalty(OldViolation,NewViolation,RequiredDecrease,OldPenalty,Scale,
NewPenalty)
```
Mode and number of proofs:
```
update_penalty(+number,+number,+number,+number,+number,-number) - one
```

---

abs_sum/2

Computes the sum of the absolute values of a numeric list.

Compilation flags:
`static`

Template:
`abs_sum(Values,Sum)`
Mode and number of proofs:
`abs_sum(+list(number),-number)` - `one`

---

validate_numeric_vector/2

Checks that a vector is a list of numbers.

Compilation flags:
`static`

Template:
`validate_numeric_vector(Vector,Domain)`
Mode and number of proofs:
`validate_numeric_vector(+list,+atom)` - `one_or_error`

Exceptions:
`Vector` is not a numeric vector:
`domain_error(Domain,Vector)`

---

validate_constraint_data/4

Checks that constraint values are numeric and that the Jacobian has one numeric row per constraint and one column per point component.

Compilation flags:
`static`

Template:
`validate_constraint_data(Point,Values,Jacobian,Domain)`
Mode and number of proofs:
`validate_constraint_data(+list(number),+list,+list,+atom)` - `one_or_error`

Exceptions:
`Values` is not a numeric vector:
`domain_error(Domain,Values)`
`Jacobian` does not have one numeric row per constraint and one column per point component:

```
domain_error(Domain,Jacobian)
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.49.5 `log_barrier(Problem,InnerSolver)`

- `Problem` - Problem object implementing `constrained_optimization_problem_protocol`. The initial point need not be strictly feasible.
- `InnerSolver` - Atom naming a local_optimization solver class taking a single problem parameter, e.g. `bfgs` or `lbfgs`. Constructed dynamically each outer iteration as `InnerSolver(SubProblem)`.

Interior-point log-barrier method for inequality constraints, optionally combined with a quadratic penalty for equality constraints.

Availability:

```
logtalk_load(constrained_optimization(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public constrained_optimization_solver
public local_optimization_solver(Problem)
```

Uses:

list

Remarks:

(none)

Inherited public predicates:

check_option/1 check_options/1 default_option/1 default_options/1 option/2 option/3 run/2 run/3 run/4 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

constrained_optimization_problem_protocol, *augmented_lagrangian(Problem,InnerSolver)*, *quadratic_penalty(Problem,InnerSolver)*, *sqp_active_set(Problem)*, *local_optimization_solver(Problem)*, *bfgs(Problem)*, *lbfgs(Problem)*

object

1.49.6 penalty_sub_problem(Problem,OuterIteration,StartPoint,Rho,Sign)

- `Problem` - The original constrained problem.
- `OuterIteration` - One-based outer iteration number.
- `StartPoint` - Point the inner solve should start from (the current outer iterate).
- `Rho` - Current penalty parameter.
- `Sign` - 1.0 to minimize the original objective, -1.0 to maximize it.

Internal quadratic-penalty subproblem: objective(x) = Sign*f(x) + (rho/2)*(||g(x)||^2 + ||max(0,h(x))||^2). Instantiated fresh by quadratic_penalty(_,_) every outer iteration, starting from StartPoint (the previous outer iterate, so each inner solve warm-starts from where the outer loop left off); not meant to be used directly.

Availability:
`logtalk_load(constrained_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *local_optimization_problem_protocol*
Imports:
`public` *constrained_optimization_solver*
Uses:
linear_algebra
list

Remarks:

(none)

Inherited public predicates:

gradient/2 hessian/2 initial_point/1 objective/2 position_bounds/1 progress/5 stop_condition/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.49.7 `phase1_sub_problem(Problem,StartPoint)`

- `Problem` - The original constrained problem.
- `StartPoint` - The point log_barrier(_,_) was actually given (Problem's own initial_point/1); phase 1 starts its search from here.

Internal phase-1 feasibility subproblem for log_barrier(_,_): minimizes sum_j max(0, h_j(x) + margin)^2 (margin = 1.0e-6), an ordinary smooth, everywhere-defined unconstrained objective, so a minimizer that reaches (near) zero gives a point with every h_j(x) < -margin, comfortably inside the barrier's domain rather than exactly on its boundary. Instantiated once, when initial_point/1 is not already strictly feasible; not meant to be used directly. See log_barrier(_,_)'s "ensure_strictly_feasible/3" for how the result is validated and how this heuristic falls back to raising domain_error/2 when it does not succeed.

Availability:

```
logtalk_load(constrained_optimization(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *local_optimization_problem_protocol*

Uses:

linear_algebra

Remarks:

(none)

Inherited public predicates:

gradient/2 *hessian/2* *initial_point/1* *objective/2* *position_bounds/1* *progress/5* *stop_condition/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.49.8 `primal_dual_interior_point(Problem)`

- `Problem` - Problem object implementing `constrained_optimization_problem_protocol` and defining `gradient/2`.

Infeasible-start primal-dual interior-point method with Mehrotra predictor-corrector for general equality, inequality, and box constraints.

Availability:

`logtalk_load(constrained_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *constrained_optimization_solver*
`public` *local_optimization_solver(Problem)*

Uses:

linear_algebra
list

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

constrained_optimization_problem_protocol, *sqp_active_set(Problem)*, *log_barrier(Problem,InnerSolver)*, *local_optimization_solver(Problem)*, *linear_algebra*

object

1.49.9 qp_active_set

Dense active-set solver for convex quadratic programs. Intended as a subroutine for sqp_active_set(_); also usable standalone for small dense QPs.

Availability:
`logtalk_load(constrained_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *qp_solver_protocol*
Imports:
`public` *constrained_optimization_solver*
Uses:
linear_algebra
list

Remarks:
(none)

Inherited public predicates:
solve/8

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

qp_solver_protocol, *sqp_active_set(Problem)*, *linear_algebra*

protocol

1.49.10 `qp_solver_protocol`

Protocol for solvers of convex quadratic programs in standard form: minimize 0.5*x^T*H*x + c^T*x subject to Aeq*x = beq and Aineq*x =< bineq. This is a numerical subroutine protocol, not a local_optimization_problem_protocol/constrained_optimization_problem_protocol solver: it operates on plain matrices and vectors, not on problem objects, since it is meant to be called once per outer/SQP iteration on a freshly linearized subproblem.

Availability:
`logtalk_load(constrained_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *solve/8*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`solve/8`

Solves min 0.5*x^T*H*x + c^T*x s.t. Aeq*x = beq, Aineq*x =< bineq, also returning the Lagrange multipliers Lambda at the solution: the first length(Beq) elements are the equality-constraint multipliers, followed by one element per row of Aineq/Bineq (in that order), each 0.0 if that inequality is inactive at X. H must be symmetric positive semi-definite over the feasible set for X to be a global minimizer. Fails (rather than raising an error) when no feasible point is found or the problem is unbounded below on the feasible set, since this predicate is meant to be used as a subroutine inside a larger iterative solver (such as sqp_active_set(_), which needs the multipliers to update its Hessian-of-the-Lagrangian approximation and merit-function penalty) that must react to that failure, not crash on it.

Compilation flags:

`static`

Template:

`solve(H,C,Aeq,Beq,Aineq,Bineq,X,Lambda)`

Mode and number of proofs:

`solve(+list(list(number)),+list(number),+list(list(number)),+list(number),+list(list(number)),+list(number),-list(number),-list(number))` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

qp_active_set, *sqp_active_set(Problem)*

object

1.49.11 `quadratic_penalty(Problem,InnerSolver)`

- `Problem` - Problem object implementing `constrained_optimization_problem_protocol`.
- `InnerSolver` - Atom naming a local_optimization solver class taking a single problem parameter, e.g. `bfgs` or `lbfgs`. Constructed dynamically each outer iteration as `InnerSolver(SubProblem)`.

Quadratic penalty method for problems with general equality and inequality constraints, delegating each penalized subproblem to an existing local optimization solver.

Availability:

`logtalk_load(constrained_optimization(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *constrained_optimization_solver*
`public` *local_optimization_solver(Problem)*

Uses:
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

constrained_optimization_problem_protocol, *augmented_lagrangian(Problem,InnerSolver)*, *sqp_active_set(Problem)*, *local_optimization_solver(Problem)*, *bfgs(Problem)*, *lbfgs(Problem)*

object

1.49.12 sqp_active_set(Problem)

- `Problem` - Problem object implementing `constrained_optimization_problem_protocol` and defining `gradient/2`.

Sequential Quadratic Programming local optimizer for problems with general equality and inequality constraints. Requires the problem to define `gradient/2`; constraint predicates and their Jacobians are optional.

Availability:

`logtalk_load(constrained_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *constrained_optimization_solver*
`public` *local_optimization_solver(Problem)*

Uses:

linear_algebra
list
qp_active_set

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

constrained_optimization_problem_protocol, *qp_solver_protocol*, *qp_active_set*, *local_optimization_solver(Problem)*, *bfgs(Problem)*

object

1.49.13 sub_problem(Problem,OuterIteration,StartPoint,EqLambda,IneqLambda,Rho,Sign)

- `Problem` - The original constrained problem.
- `OuterIteration` - One-based outer iteration number.
- `StartPoint` - Point the inner solve should start from (the current outer iterate).
- `EqLambda` - Current equality-constraint multiplier estimates (possibly []).
- `IneqLambda` - Current inequality-constraint multiplier estimates (possibly []).
- `Rho` - Current penalty parameter.
- `Sign` - 1.0 to minimize the original objective, -1.0 to maximize it.

Internal augmented-Lagrangian penalized subproblem: objective(x) = Sign*f(x) + penalty(x), where penalty adds the Hestenes-Powell term for each equality constraint and the smooth Rockafellar/Bertsekas term for each inequality constraint at the current multiplier estimates EqLambda/IneqLambda and penalty parameter Rho. Instantiated fresh by augmented_lagrangian(_,_) every outer iteration, starting from StartPoint (the previous outer iterate, not Problem's own initial_point/1, so each inner solve warm-starts from where the outer loop left off); not meant to be used directly.

Availability:

```
logtalk_load(constrained_optimization(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public local_optimization_problem_protocol
```

Imports:

```
public constrained_optimization_solver
```

Uses:

linear_algebra
list

Remarks:

(none)

Inherited public predicates:

gradient/2 *hessian/2* *initial_point/1* *objective/2* *position_bounds/1* *progress/5* *stop_condition/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.50 copeland_ranker

object

1.50.1 `copeland_ranker`

Copeland pairwise preference ranker. Learns one deterministic score per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol by comparing aggregated head-to-head outcomes for each observed matchup and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:

`logtalk_load(copeland_ranker(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-08-18

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public ranking_dataset_common
public score_ranker_model_common
```

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *bradley_terry_ranker*

1.51 core

category

1.51.1 `core_messages`

Logtalk core (compiler and runtime) default message tokenization.

Availability:
`built_in`

Author: Paulo Moura
Version: 1:150:0
Date: 2026-08-10

Compilation flags:
`static`

Provides:
logtalk::message_prefix_stream/4
logtalk::message_tokens//2

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.51.2 `expanding`

Term and goal expansion protocol.

Availability:
`built_in`

Author: Paulo Moura
Version: 1:1:0
Date: 2016-07-12

Compilation flags:
`static, built_in`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *goal_expansion/2*
  - *term_expansion/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

goal_expansion/2

Defines a goal expansion. Called recursively until a fixed point is reached on goals found while compiling a source file (except for goals wrapped using the `{}/1` compiler bypass control construct).

Compilation flags:

```
static
```

Template:

```
goal_expansion(Goal,ExpandedGoal)
```

Mode and number of proofs:

```
goal_expansion(+callable,-callable) - zero_or_one
```

---

term_expansion/2

Defines a term expansion. Called until it succeeds on all terms read while compiling a source file (except for terms skipped by using the conditional compilation directives or wrapped using the `{}/1` compiler bypass control construct).

Compilation flags:

```
static
```

Template:

```
term_expansion(Term,ExpandedTerms)
```

Mode and number of proofs:

```
term_expansion(+term,-term) - zero_or_one
term_expansion(+term,-list(term)) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

protocol

1.51.3 forwarding

Message forwarding protocol.

Availability:
`built_in`

Author: Paulo Moura
Version: 1:0:1
Date: 2025-05-15

Compilation flags:
`static, built_in`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *forward/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

forward/1

User-defined message forwarding handler, automatically called (if defined) by the runtime for any message that the receiving object does not understand.

Compilation flags:
`static`

Template:
`forward(Message)`
Mode and number of proofs:
`forward(+callable)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.51.4 `logtalk`

Built-in object providing message printing, debugging, library, source file, halt predicates, and hacking methods.

Availability:
`built_in`

Author: Paulo Moura
Version: 3:5:0
Date: 2026-05-04

Compilation flags:
`static, built_in, context_switching_calls, threaded`

Dependencies:
(none)

Remarks:

- Default message kinds: `silent`, `silent(Key)`, `banner`, `help`, `comment`, `comment(Key)`, `information`, `information(Key)`, `warning`, `warning(Key)`, `error`, `error(Key)`, `debug`, `debug(Key)`, `question`, and `question(Key)`.
- Printing of silent messages: By default, silent messages are not printed. These messages are only useful when intercepted.
- Printing of banner and comment messages: By default, banner and comment messages are only printed when the `report` flag is turned on.
- Printing of help, information, and question messages: These messages are always printed by default as they provide requested output.
- Printing of warning messages: By default, warning messages are not printed when the `report` flag is turned off.
- Printing of error messages: These messages are always printed by default.
- Printing of debug messages: By default, debug messages are only printed when the `debug` flag is turned on. The compiler suppresses debug message printing goals when compiling in optimized mode.
- Meta messages: A *meta message* is a message that have another message as argument and is typically used for debugging messages. Meta messages avoid the need of defining tokenizer rules for every message but can be intercepted as any other message.
- Meta message `@Message`: By default, the message is printed as passed to the `write/1` predicate followed by a newline.
- Meta message `Key-Value`: By default, the message is printed as "Key: Value" followed by a newline. The key is printed as passed to the `write/1` predicate while the value is printed as passed to the `writeq/1` predicate.
- Meta message `Format+Arguments`: By default, the message is printed as passed to the `format/2` predicate.

- Meta message `List`: By default, the list items are printed indented one per line. The items are preceded by a dash and can be `@Message`, `Key-Value`, or `Format+Arguments` messages. If that is not the case, the item is printed as passed to the `writeq/1` predicate.
- Meta message `Title::List`: By default, the title is printed followed by a newline and the indented list items, one per line. The items are printed as in the `List` meta message.
- Meta message `[Stream,Prefix]>>Goal`: By default, call user-defined `Goal` in the context of `user`. The use of a lambda expression allows passing the message stream and prefix. Printing the prefix is delegated to the goal.
- Meta message `[Stream]>>Goal`: By default, call user-defined `Goal` in the context of `user`. The use of a lambda expression allows passing the message stream.
- Message tokens: `at_same_line`, `tab(Expression)`, `nl`, `flush`, `Format-Arguments`, `term(Term,Options)`, `ansi(Attributes,Format,Arguments)`, `begin(Kind,Variable)`, and `end(Variable)`.
- Multi-threading applications: Predicates calling methods such as `print_message/3`, `ask_question/5`, or `compile_aux_clauses/1` may need to be declared synchronized in order to avoid race conditions.

Inherited public predicates:

(none)

- *Public predicates*
  - *print_message/3*
  - *print_message_tokens/3*
  - *print_message_token/4*
  - *message_tokens//2*
  - *message_prefix_stream/4*
  - *message_prefix_file/6*
  - *message_hook/4*
  - *ask_question/5*
  - *question_hook/6*
  - *question_prompt_stream/4*
  - *trace_event/2*
  - *debug_handler/1*
  - *active_debug_handler/1*
  - *activate_debug_handler/1*
  - *deactivate_debug_handler/0*
  - *debug_handler/3*
  - *expand_library_path/2*
  - *loaded_file/1*
  - *loaded_file_property/2*
  - *loaded_files_topological_sort/1*
  - *loaded_files_topological_sort/2*
  - *file_type_extension/2*
  - *halt/1*
  - *halt/0*
  - *halt_hook/2*
  - *compile_aux_clauses/1*
  - *entity_prefix/2*
  - *compile_predicate_heads/4*
  - *compile_predicate_indicators/3*
  - *decompile_predicate_heads/4*

    - *decompile_predicate_indicators/4*
    - *execution_context/7*
  - *Protected predicates*
  - *Private predicates*
    - *active_debug_handler_/1*
  - *Operators*

Public predicates

print_message/3

Prints a message of the given kind for the specified component.

Compilation flags:
```
static
```

Template:
```
print_message(Kind,Component,Message)
```
Mode and number of proofs:
```
print_message(+nonvar,+nonvar,+nonvar) - one
```

---

print_message_tokens/3

Print the messages tokens to the given stream, prefixing each line with the specified atom.

Compilation flags:
```
static
```

Template:
```
print_message_tokens(Stream,Prefix,Tokens)
```
Mode and number of proofs:
```
print_message_tokens(@stream_or_alias,+atom,@list(nonvar)) - one
```

---

print_message_token/4

User-defined hook predicate for printing a message token (see this object remarks).

Compilation flags:
```
dynamic, multifile
```

Template:
```
print_message_token(Stream,Prefix,Token,Tokens)
```
Mode and number of proofs:

```
print_message_token(@stream_or_alias,@atom,@nonvar,@list(nonvar)) - zero_or_one
```

---

message_tokens//2

User-defined hook grammar rule for converting a message into a list of tokens (see this object remarks).

Compilation flags:

```
dynamic, multifile
```

Template:

```
message_tokens(Message,Component)
```

Mode and number of proofs:

```
message_tokens(+nonvar,+nonvar) - zero_or_one
```

---

message_prefix_stream/4

Message line prefix and output stream to be used when printing a message given its kind and component.

Compilation flags:

```
dynamic, multifile
```

Template:

```
message_prefix_stream(Kind,Component,Prefix,Stream)
```

Mode and number of proofs:

```
message_prefix_stream(?nonvar,?nonvar,?atom,?stream_or_alias) - zero_or_more
```

---

message_prefix_file/6

Message line prefix and output file to be used when printing a message given its kind and component.

Compilation flags:

```
dynamic, multifile
```

Template:

```
message_prefix_file(Kind,Component,Prefix,File,Mode,Options)
```

Mode and number of proofs:

```
message_prefix_file(?nonvar,?nonvar,?atom,?atom,?atom,?list(compound)) -
zero_or_more
```

---

message_hook/4

User-defined hook predicate for intercepting message printing calls.

Compilation flags:
```
dynamic, multifile
```

Template:
```
message_hook(Message,Kind,Component,Tokens)
```
Mode and number of proofs:
```
message_hook(+nonvar,+nonvar,+nonvar,+list(nonvar)) - zero_or_one
```

---

ask_question/5

Asks a question and reads the answer until the check predicate is true.

Compilation flags:
```
static
```

Template:
```
ask_question(Kind,Component,Question,Check,Answer)
```
Meta-predicate template:
```
ask_question(*,*,*,1,*)
```
Mode and number of proofs:
```
ask_question(+nonvar,+nonvar,+nonvar,+callable,-term) - one
```

---

question_hook/6

User-defined hook predicate for intercepting question asking calls.

Compilation flags:
```
dynamic, multifile
```

Template:
```
question_hook(Question,Kind,Component,Tokens,Check,Answer)
```
Meta-predicate template:
```
question_hook(*,*,*,*,1,*)
```
Mode and number of proofs:
```
question_hook(+nonvar,+nonvar,+nonvar,+list(nonvar),+callable,-term) - zero_or_one
```

---

question_prompt_stream/4

Prompt and input stream to be used when asking a question given its kind and component.

Compilation flags:

```
dynamic, multifile
```

Template:

```
question_prompt_stream(Kind,Component,Prompt,Stream)
```

Mode and number of proofs:

```
question_prompt_stream(?nonvar,?nonvar,?atom,?stream_or_alias) - zero_or_more
```

---

trace_event/2

Trace event handler. The runtime calls all trace event handlers using a failure-driven loop before calling the debug event handler.

Compilation flags:

```
dynamic, multifile
```

Template:

```
trace_event(Event,ExecutionContext)
```

Mode and number of proofs:

```
trace_event(@callable,@execution_context) - zero
```

Remarks:

- Unification events: Generated after a successful unification with a fact - `fact(Entity,Fact,Clause,File,Line)` - or a rule head - `rule(Entity,Head,Clause,File,Line)`.
- Goal events: Generated when calling a goal: `top_goal(Goal,CompiledGoal)` or `goal(Goal,CompiledGoal)`.

---

debug_handler/1

Enumerates, by backtracking, all declared debug handler providers. Define a clause for this predicate to declare a new debug handler provider.

Compilation flags:

```
static, multifile
```

Template:

```
debug_handler(Provider)
```

Mode and number of proofs:

```
debug_handler(?object_identifier) - zero_or_more
debug_handler(?category_identifier) - zero_or_more
```

---

active_debug_handler/1

Current active debug handler provider if any. There is at most one active debug handler provider at any given moment.

Compilation flags:
`static`

Template:
`active_debug_handler(Provider)`
Mode and number of proofs:
`active_debug_handler(?category_identifier)` - `zero_or_one`
`active_debug_handler(?category_identifier)` - `zero_or_one`

---

activate_debug_handler/1

Activates the given debug handler provider. There is at most one active debug handler provider at any given moment. Fails if the object or category is not declared as a debug handler provider.

Compilation flags:
`static`

Template:
`activate_debug_handler(Provider)`
Mode and number of proofs:
`activate_debug_handler(@object_identifier)` - `zero_or_one`
`activate_debug_handler(@category_identifier)` - `zero_or_one`

---

deactivate_debug_handler/0

Deactivates the current debug handler provider if any.

Compilation flags:
`static`

Mode and number of proofs:
`deactivate_debug_handler` - `one`

---

debug_handler/3

Debug event handler. Called by the runtime when the given provider is active.

Compilation flags:

```
static, multifile
```

Template:

```
debug_handler(Provider,Event,ExecutionContext)
```

Mode and number of proofs:

```
debug_handler(+object_identifier,+callable,+execution_context) - zero_or_more
debug_handler(+category_identifier,+callable,+execution_context) - zero_or_more
```

Remarks:

- Unification events: Generated after a successful unification with a fact - `fact(Entity,Fact,Clause,File,Line)` - or a rule head - `rule(Entity,Head,Clause,File,Line)`.
- Goal events: Generated when calling a goal: `top_goal(Goal,CompiledGoal)` or `goal(Goal,CompiledGoal)`.

---

expand_library_path/2

Expands a library alias (an atom) or a compound term (using library notation) into its absolute path. Uses a depth bound to prevent loops.

Compilation flags:

```
static
```

Template:

```
expand_library_path(LibraryAlias,AbsolutePath)
```

Mode and number of proofs:

```
expand_library_path(+atom,?atom) - zero_or_one
expand_library_path(+callable,?atom) - zero_or_one
```

---

loaded_file/1

Enumerates, by backtracking, all loaded files, returning their full paths.

Compilation flags:

```
static
```

Template:

```
loaded_file(Path)
```

Mode and number of proofs:

```
loaded_file(?atom) - zero_or_more
```

---

loaded_file_property/2

Enumerates, by backtracking, loaded file properties.

Compilation flags:
`static`

Template:
`loaded_file_property(Path,Property)`
Mode and number of proofs:
`loaded_file_property(?atom,?compound)` - `zero_or_more`

Remarks:

- Property `basename/1`: Basename of the file (includes the file extension, if any).
- Property `directory/1`: Directory of the file (ending with a slash).
- Property `mode/1`: Compilation mode of the file (possible values are `optimal`, `normal`, and `debug`).
- Property `flags/1`: Explicit flags used for compiling the file.
- Property `text_properties/1`: List of the file text properties (`encoding/1` and `bom/1`). Empty if no `encoding/1` directive is present and the stream used for reading the file does not have a `bom/1` (or equivalent) property.
- Property `target/1`: Full path of the generated intermediate Prolog file.
- Property `modified/1`: File modification time stamp (should be regarded as an opaque but otherwise comparable term).
- Property `parent/1`: Full path of the parent file that loaded the file.
- Property `includes/2`: Full path of a file included by the file and the line of the `include/1` directive.
- Property `includes/1`: Full path of a file included by the file.
- Property `library/1`: Library alias for the library that includes the file.
- Property `object/3`: Identifier for an object defined in the file and the start and end lines of its definition.
- Property `object/1`: Identifier for an object defined in the file.
- Property `protocol/3`: Identifier for a protocol defined in the file and the start and end lines of its definition.
- Property `protocol/1`: Identifier for a protocol defined in the file.
- Property `category/3`: Identifier for a category defined in the file and the start and end lines of its definition.
- Property `category/1`: Identifier for a category defined in the file.

---

loaded_files_topological_sort/1

Returns a list of full paths for all loaded user-defined files sorted by dependencies.

Compilation flags:
`static`

Template:
`loaded_files_topological_sort(Sorted)`
Mode and number of proofs:
`loaded_files_topological_sort(--list(atom))` - `one`

---

loaded_files_topological_sort/2

Sorts a list of full paths for loaded files by dependencies.

Compilation flags:
`static`

Template:
`loaded_files_topological_sort(Paths,Sorted)`
Mode and number of proofs:
`loaded_files_topological_sort(+list(atom),--list(atom))` - `one`

---

file_type_extension/2

Enumerates, by backtracking, all defined file type extensions. The defined types are: `source`, `object`, `logtalk`, `prolog`, and `tmp`. The source type returns both `logtalk` and `prolog` type extensions.

Compilation flags:
`static`

Template:
`file_type_extension(Type,Extension)`
Mode and number of proofs:
`file_type_extension(?atom,?atom)` - `zero_or_more`

---

halt/1

Halts the Logtalk process with the given exit code. Calls all defined halt hook goals whose exit code unifies with its argument. A warning message is printed in case of goal failures or errors (which are otherwise ignored).

Compilation flags:
`static`

Template:
`halt(ExitCode)`
Mode and number of proofs:
`halt(+integer)` - `one`

---

halt/0

Halts the Logtalk process with the given exit code. Calls all defined halt hook goals whose exit code unifies with the integer zero. A warning message is printed in case of goal failures or errors (which are otherwise ignored).

Compilation flags:

```
static
```

Mode and number of proofs:

```
halt - one
```

---

halt_hook/2

Hook predicate defining a goal to be called when sending the `halt/0-1` messages.

Compilation flags:

```
dynamic, multifile
```

Template:

```
halt_hook(ExitCode,Goal)
```

Mode and number of proofs:

```
halt_hook(?var,?callable) - zero_or_more
halt_hook(?integer,?callable) - zero_or_more
```

---

compile_aux_clauses/1

Compiles a list of auxiliary clauses. Can only be called during source file compilation, usually from `term_expansion/2` or `goal_expansion/2` hook predicate definitions.

Compilation flags:

```
static
```

Template:

```
compile_aux_clauses(Clauses)
```

Mode and number of proofs:

```
compile_aux_clauses(@list(clause)) - one
```

---

entity_prefix/2

Converts between an entity identifier and the entity prefix that is used for its compiled code. When none of the arguments is instantiated, it returns the identifier and the prefix of the entity under compilation, if any.

Compilation flags:

```
static
```

Template:

```
entity_prefix(Entity,Prefix)
```

Mode and number of proofs:

```
entity_prefix(?entity_identifier,?atom) - zero_or_one
```

---

compile_predicate_heads/4

Compiles clause heads. The heads are compiled in the context of the entity under compilation when the entity argument is not instantiated.

Compilation flags:

```
static
```

Template:

```
compile_predicate_heads(Heads,Entity,CompiledHeads,ExecutionContext)
```

Mode and number of proofs:

```
compile_predicate_heads(@list(callable),?entity_identifier,-list(callable),
@execution_context) - zero_or_one
compile_predicate_heads(@conjunction(callable),?entity_identifier,
-conjunction(callable),@execution_context) - zero_or_one
compile_predicate_heads(@callable,?entity_identifier,-callable,
@execution_context) - zero_or_one
```

---

compile_predicate_indicators/3

Compiles predicate indicators. The predicate are compiled in the context of the entity under compilation when the entity argument is not instantiated.

Compilation flags:

```
static
```

Template:

```
compile_predicate_indicators(PredicateIndicators,Entity,
CompiledPredicateIndicators)
```

Mode and number of proofs:

```
compile_predicate_indicators(@list(predicate_indicator),?entity_identifier,
-list(predicate_indicator)) - zero_or_one
```

```
compile_predicate_indicators(@conjunction(predicate_indicator),?
entity_identifier,-conjunction(predicate_indicator)) - zero_or_one
compile_predicate_indicators(@predicate_indicator,?entity_identifier,
-predicate_indicator) - zero_or_one
```

---

decompile_predicate_heads/4

Decompiles clause heads. All compiled clause heads must belong to the same entity, which must be loaded.

Compilation flags:

```
static
```

Template:

```
decompile_predicate_heads(CompiledHeads,Entity,Type,Heads)
```

Mode and number of proofs:

```
decompile_predicate_heads(@list(callable),-entity_identifier,-atom,
-list(callable)) - zero_or_one
decompile_predicate_heads(@conjunction(callable),-entity_identifier,-atom,
-conjunction(callable)) - zero_or_one
decompile_predicate_heads(@callable,-entity_identifier,-atom,-callable) -
zero_or_one
```

---

decompile_predicate_indicators/4

Decompiles predicate indicators. All compiled predicate indicators must belong to the same entity, which must be loaded.

Compilation flags:

```
static
```

Template:

```
decompile_predicate_indicators(CompiledPredicateIndicators,Entity,Type,
PredicateIndicators)
```

Mode and number of proofs:

```
decompile_predicate_indicators(@list(predicate_indicator),-entity_identifier,
-atom,-list(predicate_indicator)) - zero_or_one
decompile_predicate_indicators(@conjunction(predicate_indicator),
-entity_identifier,-atom,-conjunction(predicate_indicator)) - zero_or_one
decompile_predicate_indicators(@predicate_indicator,-entity_identifier,-atom,
-predicate_indicator) - zero_or_one
```

---

execution_context/7

Execution context term data. Execution context terms should be considered opaque terms subject to change without notice.

Compilation flags:
`static`

Template:
`execution_context(ExecutionContext,Entity,Sender,This,Self,MetaCallContext,CoinductionStack)`

Mode and number of proofs:
`execution_context(?nonvar,?entity_identifier,?object_identifier,?object_identifier,?object_identifier,@list(callable),@list(callable))` - `zero_or_one`

Protected predicates

(none)

Private predicates

active_debug_handler_/1

Current active debug handler provider. There is at most one active debug handler provider at any given moment.

Compilation flags:
`dynamic`

Template:
`active_debug_handler_(Provider)`

Mode and number of proofs:
`active_debug_handler_(?entity_identifier)` - `zero_or_one`

Operators

(none)

protocol

1.51.5 monitoring

Event handlers protocol. The handlers are automatically called by the runtime for messages sent using the `::/2` control construct from objects or categories compiled with the `events` flag set to `allow`.

Availability:
`built_in`

Author: Paulo Moura

Version: 1:2:0
Date: 2018-11-29

Compilation flags:
`static, built_in`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *before/3*
  - *after/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

before/3

Event handler for `before` events. A `before` event handler may prevent a method from being looked up or called by failing.

Compilation flags:
`static`

Template:
`before(Object,Message,Sender)`
Mode and number of proofs:
`before(?term,?term,?term)` - `zero_or_more`

---

after/3

Event handler for `after` events. An `after` event handler may prevent a method from succeeding by failing.

Compilation flags:
`static`

Template:
`after(Object,Message,Sender)`

Mode and number of proofs:
`after(?term,?term,?term)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.51.6 `user`

Pseudo-object representing the plain Prolog database. Can be used as a monitor by defining `before/3` and `after/3` predicates. Can be used as a hook object by defining `term_expansion/2` and `goal_expansion/2` multifile and dynamic predicates.

Availability:
`built_in`

Author: Paulo Moura
Version: 1:6:0
Date: 2024-11-11

Compilation flags:
`static, built_in, context_switching_calls, dynamic_declarations, threaded`

Implements:
`public` *expanding*
`public` *forwarding*
`public` *monitoring*
Uses:
user

Remarks:
(none)

Inherited public predicates:
after/3 *before/3* *forward/1* *goal_expansion/2* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.52 coroutining

object

1.52.1 `coroutining`

Coroutining predicates.

Availability:
`logtalk_load(coroutining(loader))`

Author: Paulo Moura
Version: 0:6:1
Date: 2026-10-30

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Supported backend Prolog systems: ECLiPSe, SICStus Prolog, Shumway Prolog, SWI-Prolog, Trealla Prolog, XVM, and YAP.

Inherited public predicates:
(none)

- *Public predicates*
  - *dif/2*
  - *dif/1*
  - *freeze/2*

    - *frozen/2*
    - *when/2*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

dif/2

Sets a constraint that is true iff the two terms are different.

Compilation flags:
```
static
```

Template:
```
dif(Term1,Term2)
```
Mode and number of proofs:
```
dif(+term,+term) - zero_or_one
```

---

dif/1

Sets a set of constraints that are true iff all terms in a list are different.

Compilation flags:
```
static
```

Template:
```
dif(Terms)
```
Mode and number of proofs:
```
dif(+list(term)) - zero_or_one
```

---

freeze/2

Delays the execution of a goal until a variable is bound.

Compilation flags:
```
static
```

Template:
```
freeze(Variable,Goal)
```
Meta-predicate template:
```
freeze(*,0)
```

Mode and number of proofs:
`freeze(+term,+callable)` - `zero_or_more`

`frozen/2`

Unifies `Goal` with the goal delayed by `Variable`. When no goals are frozen on `Variable`, `Goal` is unified with true.

Compilation flags:
`static`

Template:
`frozen(Variable,Goal)`
Mode and number of proofs:
`frozen(@var,--callable)` - `one`

`when/2`

Calls `Goal` when `Condition` becomes true. The portable conditions are: `nonvar/1`, `ground/1`, `(,)/2`, and `(;)/2`.

Compilation flags:
`static`

Template:
`when(Condition,Goal)`
Meta-predicate template:
`when(*,0)`
Mode and number of proofs:
`when(+callable,+callable)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.53 crontab

object

1.53.1 `crontab`

Parser and generator for user and system crontab files using atoms for text representation.

Availability:
`logtalk_load(crontab(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *crontab(atom)*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *entry/1* *generate/2* *generate/3* *option/2* *option/3* *parse/2* *parse/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.53.2 `crontab(Representation)`

- `Representation` - Text representation. Valid values are `atom`, `chars`, and `codes`.

Parser and generator for user and system crontab files.

Availability:
`logtalk_load(crontab(loader))`

Author: Paulo Moura
Version: 1:1:2
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
list
reader
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *parse/2*
  - *parse/3*
  - *generate/2*
  - *generate/3*
  - *entry/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a user crontab from the given source using the default options.

Compilation flags:
`static`

Template:
`parse(Source,Entries)`
Mode and number of proofs:
`parse(++compound,--list)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid JSON source term:
`domain_error(json_source,Source)`
`Source` is a valid JSON source term but does not contain parsable JSON:
`domain_error(json,Source)`

---

parse/3

Parses a crontab from the given source. The supported option is `format(user)` (default) or `format(system)`.

Compilation flags:
`static`

Template:
`parse(Source,Entries,Options)`
Mode and number of proofs:
`parse(++compound,--list,+list(compound))` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid JSON source term:
`domain_error(json_source,Source)`
`Source` is a valid JSON source term but does not contain parsable JSON:
`domain_error(json,Source)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

generate/2

Generates a user crontab to the given sink using the default options.

Compilation flags:
`static`

Template:
`generate(Sink,Entries)`
Mode and number of proofs:
`generate(+compound,+list)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid JSON sink term:
`domain_error(json_sink,Sink)`
A JSON version of `Term` cannot be generated by the implementation:
`domain_error(json_term,Term)`

---

generate/3

Generates a crontab to the given sink. The supported option is `format(user)` (default) or `format(system)`.

Compilation flags:
`static`

Template:
`generate(Sink,Entries,Options)`
Mode and number of proofs:
`generate(+compound,+list,+list(compound))` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid JSON sink term:
`domain_error(json_sink,Sink)`
A JSON version of `Term` cannot be generated by the implementation:
`domain_error(json_term,Term)`
`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`entry/1`

True if the argument is a valid crontab entry using the current text representation.

Compilation flags:

`static`

Template:

`entry(Entry)`

Mode and number of proofs:

`entry(@nonvar)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.54 crs_projections

object

1.54.1 crs_projections

Coordinate reference systems and common transformations for 2D and 3D WGS84 geodetic coordinates, Web Mercator, World Mercator, ECEF, local ENU tangent-plane, Lambert azimuthal equal-area, and UTM coordinates.

Availability:
`logtalk_load(crs_projections(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-10

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *crs_projections_protocol*
Uses:
geospatial

Remarks:

- Geographic coordinates: WGS84 coordinates are represented as `geographic(Latitude,Longitude)` in degrees.
- 3D geodetic coordinates: WGS84 3D coordinates are represented as `geographic(Latitude,Longitude,EllipsoidalHeight)` using degrees for angular axes and meters for height.
- Projected coordinates: Web Mercator, World Mercator, and Lambert azimuthal equal-area coordinates are represented as meter-based `projected(X,Y)` pairs. UTM coordinates are represented as `grid(Easting,Northing)` pairs.
- Geocentric coordinates: ECEF coordinates are represented as `ecef(X,Y,Z)` triples in meters.
- Local tangent-plane coordinates: ENU coordinates are represented as `enu(East,North,Up)` triples in meters relative to an origin coordinate.
- UTM coverage: UTM helpers and transformations only succeed for latitudes in the `[-80.0,84.0[` range.

Inherited public predicates:
crs_dimensions/2 *crs_epsg/2* *crs_kind/2* *crs_name/2* *crs_units/2* *epsg_crs/2* *lambert_azimuthal_equal_area/3* *lambert_azimuthal_equal_area_inverse/3* *local_tangent_plane/3* *local_tangent_plane_inverse/3* *transform/4* *utm_crs/2* *utm_zone/2* *valid_coordinate/2* *valid_crs/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

crs_projections_protocol, *geospatial*

protocol

1.54.2 `crs_projections_protocol`

Coordinate reference systems and coordinate transformation predicates protocol.

Availability:
`logtalk_load(crs_projections(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-10

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *valid_crs/1*
  - *crs_name/2*
  - *crs_kind/2*
  - *crs_units/2*

      - *crs_dimensions/2*
      - *crs_epsg/2*
      - *epsg_crs/2*
      - *valid_coordinate/2*
      - *local_tangent_plane/3*
      - *local_tangent_plane_inverse/3*
      - *lambert_azimuthal_equal_area/3*
      - *lambert_azimuthal_equal_area_inverse/3*
      - *utm_zone/2*
      - *utm_crs/2*
      - *transform/4*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

valid_crs/1

True when the argument is a supported coordinate reference system. Supported values are `wgs84`, `wgs84_3d`, `web_mercator`, `world_mercator`, `ecef`, `enu(OriginCoordinate)`, `lambert_azimuthal_equal_area(OriginCoordinate)`, and `utm(Zone,Hemisphere)` terms.

Compilation flags:

```
static
```

Template:

```
valid_crs(CRS)
```

Mode and number of proofs:

```
valid_crs(@term) - zero_or_one
```

---

crs_name/2

Maps a supported coordinate reference system to a human-readable name.

Compilation flags:

```
static
```

Template:

```
crs_name(CRS,Name)
```

Mode and number of proofs:

```
crs_name(+term,-atom) - zero_or_one
```

---

crs_kind/2

Classifies a supported coordinate reference system as `geographic`, `projected`, `geocentric`, or `local_tangent_plane`.

Compilation flags:
`static`

Template:
`crs_kind(CRS,Kind)`
Mode and number of proofs:
`crs_kind(+term,-atom)` - `zero_or_one`

---

crs_units/2

Returns the coordinate axis units used by a supported coordinate reference system. Supported values are `degrees`, `meters`, and `[degrees,degrees,meters]`.

Compilation flags:
`static`

Template:
`crs_units(CRS,Units)`
Mode and number of proofs:
`crs_units(+term,-term)` - `zero_or_one`

---

crs_dimensions/2

Returns the coordinate dimensionality for a supported coordinate reference system.

Compilation flags:
`static`

Template:
`crs_dimensions(CRS,Dimensions)`
Mode and number of proofs:
`crs_dimensions(+term,-positive_integer)` - `zero_or_one`

---

crs_epsg/2

Maps a supported coordinate reference system to the corresponding EPSG code.

Compilation flags:
`static`

Template:
`crs_epsg(CRS,EPSG)`
Mode and number of proofs:
`crs_epsg(+term,-positive_integer)` - `zero_or_one`

---

epsg_crs/2

Maps a supported EPSG code to the corresponding coordinate reference system term.

Compilation flags:
`static`

Template:
`epsg_crs(EPSG,CRS)`
Mode and number of proofs:
`epsg_crs(+positive_integer,-term)` - `zero_or_one`

---

valid_coordinate/2

True when the coordinate is valid for the given coordinate reference system. `wgs84` coordinates are represented as `geographic(Latitude,Longitude)` in degrees, `wgs84_3d` coordinates as `geographic(Latitude,Longitude,EllipsoidalHeight)` in degrees and meters, `web_mercator`, `world_mercator`, and `lambert_azimuthal_equal_area(OriginCoordinate)` coordinates as `projected(X,Y)` pairs in meters, `ecef` coordinates as `ecef(X,Y,Z)` triples in meters, `enu(OriginCoordinate)` coordinates as `enu(East,North,Up)` triples in meters, and `utm(Zone,Hemisphere)` coordinates as `grid(Easting,Northing)` pairs in meters.

Compilation flags:
`static`

Template:
`valid_coordinate(CRS,Coordinate)`
Mode and number of proofs:
`valid_coordinate(+term,@compound)` - `zero_or_one`

---

local_tangent_plane/3

Projects a WGS84 geographic coordinate to local `enu(East,North,Up)` coordinates in meters using an origin coordinate. For height-preserving 3D geodetic workflows, use `transform/4` with `wgs84_3d`.

Compilation flags:

`static`

Template:

`local_tangent_plane(Origin,Coordinate,LocalCoordinate)`

Mode and number of proofs:

`local_tangent_plane(+compound,+compound,-compound)` - `zero_or_one`

---

local_tangent_plane_inverse/3

Converts local `enu(East,North,Up)` coordinates in meters back to a WGS84 geographic coordinate using an origin coordinate. For height-preserving 3D geodetic workflows, use `transform/4` with `wgs84_3d`.

Compilation flags:

`static`

Template:

`local_tangent_plane_inverse(Origin,LocalCoordinate,Coordinate)`

Mode and number of proofs:

`local_tangent_plane_inverse(+compound,+compound,-compound)` - `zero_or_one`

---

lambert_azimuthal_equal_area/3

Projects a WGS84 geographic coordinate to a Lambert azimuthal equal-area plane centered on an origin coordinate, returning `projected(X,Y)` coordinates in meters.

Compilation flags:

`static`

Template:

`lambert_azimuthal_equal_area(Origin,Coordinate,ProjectedCoordinate)`

Mode and number of proofs:

`lambert_azimuthal_equal_area(+compound,+compound,-compound)` - `zero_or_one`

---

lambert_azimuthal_equal_area_inverse/3

Converts Lambert azimuthal equal-area `projected(X,Y)` coordinates back to a WGS84 geographic coordinate using an origin coordinate.

Compilation flags:
`static`

Template:
`lambert_azimuthal_equal_area_inverse(Origin,ProjectedCoordinate,Coordinate)`
Mode and number of proofs:
`lambert_azimuthal_equal_area_inverse(+compound,+compound,-compound)` - `zero_or_one`

---

utm_zone/2

Infers the UTM longitudinal zone for a WGS84 `geographic(Latitude,Longitude)` coordinate. Uses the standard Norway and Svalbard special-zone rules and only succeeds for coordinates within the UTM latitude coverage.

Compilation flags:
`static`

Template:
`utm_zone(Coordinate,Zone)`
Mode and number of proofs:
`utm_zone(+compound,-integer)` - `zero_or_one`

---

utm_crs/2

Infers the native UTM coordinate reference system term `utm(Zone,Hemisphere)` for a WGS84 `geographic(Latitude,Longitude)` coordinate.

Compilation flags:
`static`

Template:
`utm_crs(Coordinate,CRS)`
Mode and number of proofs:
`utm_crs(+compound,-compound)` - `zero_or_one`

---

transform/4

Transforms a coordinate between supported coordinate reference systems. Transformations between projected systems are computed by converting through `wgs84`.

Compilation flags:
`static`

Template:
`transform(SourceCRS,TargetCRS,Coordinate,TransformedCoordinate)`
Mode and number of proofs:
`transform(+term,+term,+compound,-compound)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

crs_projections, *geospatial_protocol*

1.55 crypto

category

1.55.1 authenticated_channel

Identity-authenticated ephemeral X25519 channels with directional XChaCha20-Poly1305 keys, strict counters, and per-message symmetric key ratcheting.

Availability:
`logtalk_load(crypto(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static`

Complements:

crypto

Uses:

list
sha256
type

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - `authenticated_channel_initiate/4`
  - `authenticated_channel_accept/5`
  - `authenticated_channel_finalize/3`
  - `authenticated_channel_encrypt/5`
  - `authenticated_channel_decrypt/5`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`authenticated_channel_initiate/4`

Creates a signed ephemeral-key Offer and PendingState for an initiator using IdentitySeed and the pinned ResponderIdentityPublicKey.

Compilation flags:

`static`

Template:

`authenticated_channel_initiate(IdentitySeed,ResponderIdentityPublicKey,Offer,PendingState)`

Mode and number of proofs:

`authenticated_channel_initiate(+list(byte),+list(byte),-compound,-compound)` - `one_or_error`

Exceptions:

`IdentitySeed` is a partial list or a list with an element which is a variable:
`instantiation_error`
`IdentitySeed` is neither a variable nor a list of 32 bytes:
`type_error(list(byte,32),IdentitySeed)`
`IdentitySeed` contains a non-integer byte:
`type_error(integer,Byte)`
`IdentitySeed` contains an integer outside the byte range:
`domain_error(byte,Byte)`

`ResponderIdentityPublicKey` is a partial list or a list with an element which is a variable:
`instantiation_error`
`ResponderIdentityPublicKey` is neither a variable nor a list of 32 bytes:
`type_error(list(byte,32),ResponderIdentityPublicKey)`
`ResponderIdentityPublicKey` contains a non-integer byte:
`type_error(integer,Byte)`
`ResponderIdentityPublicKey` contains an integer outside the byte range:
`domain_error(byte,Byte)`

`authenticated_channel_accept/5`

Verifies Offer against the pinned InitiatorIdentityPublicKey, creates a signed Response, and returns the responder Channel state.

Compilation flags:
`static`

Template:
`authenticated_channel_accept(IdentitySeed,InitiatorIdentityPublicKey,Offer,Response,Channel)`

Mode and number of proofs:
`authenticated_channel_accept(+list(byte),+list(byte),+compound,-compound,-compound)` - `zero_or_one_or_error`

Exceptions:
`IdentitySeed` is a partial list or a list with an element which is a variable:
`instantiation_error`
`IdentitySeed` is neither a variable nor a list of 32 bytes:
`type_error(list(byte,32),IdentitySeed)`
`IdentitySeed` contains a non-integer byte:
`type_error(integer,Byte)`
`IdentitySeed` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`InitiatorIdentityPublicKey` is a partial list or a list with an element which is a variable:
`instantiation_error`
`InitiatorIdentityPublicKey` is neither a variable nor a list of 32 bytes:
`type_error(list(byte,32),InitiatorIdentityPublicKey)`
`InitiatorIdentityPublicKey` contains a non-integer byte:
`type_error(integer,Byte)`
`InitiatorIdentityPublicKey` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Offer` is a variable or contains a partial byte list or a byte list with an element which is a variable:
`instantiation_error`
`Offer` is neither a variable nor a compound term:
`type_error(compound,Offer)`
`Offer` is not a valid authenticated channel offer term:
`domain_error(authenticated_channel_offer,Offer)`
`Offer` contains a key field which is not a list of 32 bytes:

`type_error(list(byte,32),Bytes)`
`Offer` contains a signature field which is not a list of 64 bytes:
`type_error(list(byte,64),Signature)`
`Offer` contains a non-integer byte:
`type_error(integer,Byte)`
`Offer` contains an integer outside the byte range:
`domain_error(byte,Byte)`

`authenticated_channel_finalize/3`

Verifies Response against the identity pinned in PendingState and returns the initiator Channel state.

Compilation flags:
`static`

Template:
`authenticated_channel_finalize(PendingState,Response,Channel)`
Mode and number of proofs:
`authenticated_channel_finalize(+compound,+compound,-compound)` - `zero_or_one_or_error`

Exceptions:
`PendingState` is a variable or contains a partial byte list or a byte list with an element which is a variable:
`instantiation_error`
`PendingState` is neither a variable nor a compound term:
`type_error(compound,PendingState)`
`PendingState` is not a valid authenticated channel pending-state term:
`domain_error(authenticated_channel_pending_state,PendingState)`
`PendingState` contains a key field which is not a list of 32 bytes:
`type_error(list(byte,32),Bytes)`
`PendingState` contains a signature field which is not a list of 64 bytes:
`type_error(list(byte,64),Signature)`
`PendingState` contains a non-integer byte:
`type_error(integer,Byte)`
`PendingState` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Response` is a variable or contains a partial byte list or a byte list with an element which is a variable:
`instantiation_error`
`Response` is neither a variable nor a compound term:
`type_error(compound,Response)`
`Response` is not a valid authenticated channel response term:
`domain_error(authenticated_channel_response,Response)`
`Response` contains a key field which is not a list of 32 bytes:
`type_error(list(byte,32),Bytes)`
`Response` contains a signature field which is not a list of 64 bytes:
`type_error(list(byte,64),Signature)`
`Response` contains a non-integer byte:

`type_error(integer,Byte)`
`Response` contains an integer outside the byte range:
`domain_error(byte,Byte)`

`authenticated_channel_encrypt/5`

Encrypts and authenticates Plaintext and returns Message and the replacement Channel state. The caller must discard the input state.

Compilation flags:
`static`

Template:
`authenticated_channel_encrypt(Channel0,AAD,Plaintext,Message,Channel)`
Mode and number of proofs:
`authenticated_channel_encrypt(+compound,+list(byte),+list(byte),-compound,-compound)` - `one_or_error`

Exceptions:
`Channel0` is a variable or contains a partial byte list, a byte list with an element which is a variable, or a variable counter:
`instantiation_error`
`Channel0` is neither a variable nor a compound term:
`type_error(compound,Channel0)`
`Channel0` is not a valid authenticated channel state term:
`domain_error(authenticated_channel_state,Channel0)`
`Channel0` contains a hash or key field which is not a list of 32 bytes:
`type_error(list(byte,32),Bytes)`
`Channel0` contains a nonce-prefix field which is not a list of 16 bytes:
`type_error(list(byte,16),Bytes)`
`Channel0` contains a non-integer byte:
`type_error(integer,Byte)`
`Channel0` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Channel0` contains a counter which is not an integer:
`type_error(integer,Counter)`
`Channel0` contains a negative counter:
`domain_error(non_negative_integer,Counter)`
`AAD` is a partial list or a list with an element which is a variable:
`instantiation_error`
`AAD` is neither a variable nor a list of bytes:
`type_error(list(byte),AAD)`
`AAD` contains a non-integer byte:
`type_error(integer,Byte)`
`AAD` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Plaintext` is a partial list or a list with an element which is a variable:
`instantiation_error`
`Plaintext` is neither a variable nor a list of bytes:

`type_error(list(byte),Plaintext)`
`Plaintext` contains a non-integer byte:
`type_error(integer,Byte)`
`Plaintext` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Channel0` send counter is exhausted:
`resource_error(authenticated_channel_counter)`

---

authenticated_channel_decrypt/5

Authenticates and decrypts the next strictly ordered Message and returns Plaintext and the replacement Channel state. The caller must discard the input state.

Compilation flags:
`static`

Template:
`authenticated_channel_decrypt(Channel0,AAD,Message,Plaintext,Channel)`
Mode and number of proofs:
`authenticated_channel_decrypt(+compound,+list(byte),+compound,-list(byte),-compound)` - `zero_or_one_or_error`

Exceptions:
`Channel0` is a variable or contains a partial byte list, a byte list with an element which is a variable, or a variable counter:
`instantiation_error`
`Channel0` is neither a variable nor a compound term:
`type_error(compound,Channel0)`
`Channel0` is not a valid authenticated channel state term:
`domain_error(authenticated_channel_state,Channel0)`
`Channel0` contains a hash or key field which is not a list of 32 bytes:
`type_error(list(byte,32),Bytes)`
`Channel0` contains a nonce-prefix field which is not a list of 16 bytes:
`type_error(list(byte,16),Bytes)`
`Channel0` contains a non-integer byte:
`type_error(integer,Byte)`
`Channel0` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Channel0` contains a counter which is not an integer:
`type_error(integer,Counter)`
`Channel0` contains a negative counter:
`domain_error(non_negative_integer,Counter)`
`AAD` is a partial list or a list with an element which is a variable:
`instantiation_error`
`AAD` is neither a variable nor a list of bytes:
`type_error(list(byte),AAD)`
`AAD` contains a non-integer byte:
`type_error(integer,Byte)`

`AAD` contains an integer outside the byte range:
```
domain_error(byte,Byte)
```
`Message` is a variable or contains a partial byte list, a byte list with an element which is a variable, or a variable counter:
```
instantiation_error
```
`Message` is neither a variable nor a compound term:
```
type_error(compound,Message)
```
`Message` is not a valid authenticated channel message term:
```
domain_error(authenticated_channel_message,Message)
```
`Message` counter is not an integer:
```
type_error(integer,Counter)
```
`Message` counter is negative:
```
domain_error(non_negative_integer,Counter)
```
`Message` ciphertext and tag is not a list of bytes:
```
type_error(list(byte),CiphertextAndTag)
```
`Message` ciphertext and tag contains a non-integer byte:
```
type_error(integer,Byte)
```
`Message` ciphertext and tag contains an integer outside the byte range:
```
domain_error(byte,Byte)
```
`Message` ciphertext and tag is shorter than 16 bytes:
```
domain_error(minimum_byte_length(16),CiphertextAndTag)
```
`Channel0` receive counter is exhausted:
```
resource_error(authenticated_channel_counter)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.55.2 `crypto`

Transport-neutral cryptographic helper predicates.

Availability:
```
logtalk_load(crypto(loader))
```

Author: Paulo Moura
Version: 1:3:0
Date: 2026-10-12

Compilation flags:

```
static, context_switching_calls, complements(restrict)
```

Uses:

base64url_no_padding
fast_random(Algorithm)
hmac
list
md5
os
type

Remarks:

(none)

Inherited public predicates:

authenticated_channel_accept/5 *authenticated_channel_decrypt/5* *authenticated_channel_encrypt/5* *authenticated_channel_finalize/3* *authenticated_channel_initiate/4* *chacha20_block/4* *chacha20_encrypt/5* *ed25519_keypair/2* *ed25519_public_key/2* *ed25519_sign/3* *ed25519_verify/3* *x25519_keypair/2* *x25519_public_key/2* *x25519_shared_secret/3* *xchacha20/4* *xchacha20_poly1305_decrypt/5* *xchacha20_poly1305_encrypt/5* *xchacha20_subkey_and_nonce/4*

- *Public predicates*
  - *random_bytes/2*
  - *token_hex/2*
  - *token_urlsafe/2*
  - *random_below/2*
  - *hex_bytes/2*
  - *secure_compare/2*
  - *hkdf/5*
  - *pbkdf2/6*
  - *apr1/3*
  - *bcrypt/4*
  - *password_hash/4*
  - *password_hash_needs_rehash/3*
  - *verify_password_hash/2*
  - *password_hash_atom/2*
- *Protected predicates*
- *Private predicates*
  - *xor_bytes/3*
  - *constant_time_equal/2*
- *Operators*

Public predicates

random_bytes/2

Returns a list with the requested number of random bytes. It tries /dev/urandom first and falls back to a pseudo-random generator when necessary.

Compilation flags:
`static`

Template:
`random_bytes(Count,Bytes)`
Mode and number of proofs:
`random_bytes(+non_negative_integer,-list(byte))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but is less than zero:
`domain_error(non_negative_integer,Count)`

---

token_hex/2

Returns a lowercase hexadecimal token generated from the requested number of random bytes.

Compilation flags:
`static`

Template:
`token_hex(Count,Token)`
Mode and number of proofs:
`token_hex(+non_negative_integer,-atom)` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but is less than zero:
`domain_error(non_negative_integer,Count)`

---

token_urlsafe/2

Returns an unpadded Base64URL token generated from the requested number of random bytes.

Compilation flags:
`static`

Template:
`token_urlsafe(Count,Token)`
Mode and number of proofs:
`token_urlsafe(+non_negative_integer,-atom)` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but is less than zero:
`domain_error(non_negative_integer,Count)`

---

random_below/2

Returns a uniformly distributed random integer greater than or equal to zero and less than the given exclusive upper bound.

Compilation flags:
`static`

Template:
`random_below(UpperBound,Integer)`
Mode and number of proofs:
`random_below(+positive_integer,-non_negative_integer)` - `one_or_error`

Exceptions:
`UpperBound` is a variable:
`instantiation_error`
`UpperBound` is neither a variable nor an integer:
`type_error(integer,UpperBound)`
`UpperBound` is an integer but is not positive:
`domain_error(positive_integer,UpperBound)`

---

hex_bytes/2

Relates a hexadecimal atom with the corresponding list of bytes, generating lowercase hexadecimal atoms when converting from bytes.

Compilation flags:
`static`

Template:
`hex_bytes(Hex,Bytes)`
Mode and number of proofs:
`hex_bytes(+atom,-list(byte))` - `zero_or_one_or_error`
`hex_bytes(-atom,+list(byte))` - `zero_or_one_or_error`

Exceptions:
`Hex` and `Bytes` are both variables:
`instantiation_error`
`Hex` is neither a variable nor an atom:
`type_error(atom,Hex)`
`Hex` is an atom but not a valid hexadecimal atom:
`domain_error(hexadecimal_atom,Hex)`
`Bytes` is neither a variable nor a list of bytes:
`type_error(list(byte),Bytes)`
`Bytes` contains a variable byte:
`instantiation_error`
`Bytes` contains a non-integer byte:
`type_error(integer,Byte)`
`Bytes` contains an integer outside the byte range:
`domain_error(byte,Byte)`

---

secure_compare/2

Succeeds when both inputs are equal using a constant-time comparison strategy for byte sequences of the same representation.

Compilation flags:
`static`

Template:
`secure_compare(Expected,Candidate)`
Mode and number of proofs:
`secure_compare(+list(byte),+list(byte))` - `zero_or_one_or_error`
`secure_compare(+atom,+atom)` - `zero_or_one_or_error`

Exceptions:
`Expected` is a partial list or a list with an element which is a variable:
`instantiation_error`
`Expected` is neither an atom nor a list of bytes:

`type_error(list(byte),Expected)`

`Expected` is an atom and `Candidate` is not an atom:

`type_error(atom,Candidate)`

`Candidate` is a partial list or a list with an element which is a variable:

`instantiation_error`

`Candidate` is neither an atom nor a list of bytes when `Expected` is a byte list:

`type_error(list(byte),Candidate)`

`Expected` or `Candidate` contains a non-integer byte:

`type_error(integer,Byte)`

`Expected` or `Candidate` contains an integer outside the byte range:

`domain_error(byte,Byte)`

`hkdf/5`

Derives a byte sequence of the requested length from input keying material using HKDF with a hash object implementing the `hash_digest_protocol` protocol.

Compilation flags:

`static`

Template:

`hkdf(Hash,KeyMaterial,Length,Bytes,Options)`

Mode and number of proofs:

`hkdf(+object_identifier,+list(byte),+non_negative_integer,-list(byte),+list(compound))` - `one_or_error`

Exceptions:

`Hash` is a variable:

`instantiation_error`

`Hash` is not an object implementing the `hash_digest_protocol` protocol:

`domain_error(crypto_hash,Hash)`

`KeyMaterial` is a partial list or a list with an element which is a variable:

`instantiation_error`

`KeyMaterial` is a list but not a list of bytes:

`type_error(list(byte),KeyMaterial)`

`KeyMaterial` contains a non-integer byte:

`type_error(integer,Byte)`

`KeyMaterial` contains an integer outside the byte range:

`domain_error(byte,Byte)`

`Length` is a variable:

`instantiation_error`

`Length` is neither a variable nor an integer:

`type_error(integer,Length)`

`Length` is less than zero:

`domain_error(non_negative_integer,Length)`

`Length` exceeds the maximum HKDF output length for the selected hash:

`domain_error(hkdf_output_length(0,MaxLength),Length)`

`Options` is a partial list or a list with an element which is a variable:
`instantiation_error`
`Options` is neither a variable nor a list of compound terms:
`type_error(list(compound),Options)`
`Options` contains an invalid option term:
`domain_error(hkdf_option,Option)`
`Options` contains a `salt/1` or `info/1` value that is not a list of bytes:
`type_error(list(byte),Bytes)`
`Options` contains a `salt/1` or `info/1` value with a variable byte:
`instantiation_error`
`Options` contains a `salt/1` or `info/1` value with a non-integer byte:
`type_error(integer,Byte)`
`Options` contains a `salt/1` or `info/1` value with an integer outside the byte range:
`domain_error(byte,Byte)`

Remarks:

- Repeated options: When the same HKDF option is given multiple times, the last occurrence is used.
- Option `salt(Bytes)`: Uses the given byte list as the HKDF salt. When this option is absent, the salt defaults to a zero-filled byte list with the selected hash digest size.
- Option `info(Bytes)`: Uses the given byte list as the HKDF context information. When this option is absent, the context information defaults to the empty list.

---

pbkdf2/6

Derives a key from a password byte sequence and a salt using PBKDF2 with a hash object implementing the `hash_digest_protocol` protocol.

Compilation flags:
`static`

Template:
`pbkdf2(Hash,Password,Salt,Iterations,Length,DerivedKey)`
Mode and number of proofs:
`pbkdf2(+object_identifier,+list(byte),+list(byte),+integer,+positive_integer,-list(byte))` - `one_or_error`

Exceptions:
`Hash` is a variable:
`instantiation_error`
`Hash` is not an object implementing the `hash_digest_protocol` protocol:
`domain_error(crypto_hash,Hash)`
`Password` is a partial list or a list with an element which is a variable:
`instantiation_error`
`Password` is a list but not a list of bytes:
`type_error(list(byte),Password)`
`Password` contains a non-integer element:
`type_error(integer,Byte)`
`Password` contains an integer outside the byte range:

```
domain_error(byte,Byte)
```
`Salt` is a partial list or a list with an element which is a variable:
```
instantiation_error
```
`Salt` is a liust but not a list of bytes:
```
type_error(list(byte),Salt)
```
`Salt` contains a non-integer element:
```
type_error(integer,Byte)
```
`Salt` contains an integer outside the byte range:
```
domain_error(byte,Byte)
```
`Iterations` is a variable:
```
instantiation_error
```
`Iterations` is neither a variable nor an integer:
```
type_error(integer,Iterations)
```
`Iterations` is an integer but not a positive integer:
```
domain_error(positive_integer,Iterations)
```
`Length` is a variable:
```
instantiation_error
```
`Length` is neither a variable nor an integer:
```
type_error(integer,Length)
```
`Length` is an integer but is not positive:
```
domain_error(positive_integer,Length)
```
`Length` exceeds the maximum PBKDF2 output length:
```
domain_error(pbkdf2_output_length,Length)
```

---

apr1/3

Computes the Apache APR1 encoded checksum for a password byte sequence and salt byte sequence.

Compilation flags:
```
static
```

Template:
```
apr1(Password,Salt,Checksum)
```
Mode and number of proofs:
```
apr1(+list(byte),+list(byte),-list(byte)) - one_or_error
```

Exceptions:
`Password` is a partial list or a list with an element which is a variable:
```
instantiation_error
```
`Password` is a list but not a list of bytes:
```
type_error(list(byte),Password)
```
`Password` contains a non-integer element:
```
type_error(integer,Byte)
```
`Password` contains an integer outside the byte range:
```
domain_error(byte,Byte)
```
`Salt` is a partial list or a list with an element which is a variable:
```
instantiation_error
```

`Salt` is a list but not a list of bytes:
`type_error(list(byte),Salt)`
`Salt` contains a non-integer element:
`type_error(integer,Byte)`
`Salt` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Salt` is not a valid APR1 salt:
`domain_error(apr1_salt,Salt)`

---

bcrypt/4

Computes a bcrypt version 2b encoded checksum for a password, cost, and raw 16-byte salt.

Compilation flags:
`static`

Template:
`bcrypt(Password,Cost,Salt,Checksum)`
Mode and number of proofs:
`bcrypt(+list(byte),+integer,+list(byte),-list(byte))` - `one_or_error`

Exceptions:
`Password` is a partial list or a list with an element which is a variable:
`instantiation_error`
`Password` is a list but not a list of bytes:
`type_error(list(byte),Password)`
`Password` contains a non-integer element:
`type_error(integer,Byte)`
`Password` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Password` contains more than 72 bytes:
`domain_error(bcrypt_password_length,Password)`
`Cost` is a variable:
`instantiation_error`
`Cost` is not an integer:
`type_error(integer,Cost)`
`Cost` is outside the range 4 through 31:
`domain_error(bcrypt_cost,Cost)`
`Salt` is a partial list or a list with an element which is a variable:
`instantiation_error`
`Salt` is a list but not a list of bytes:
`type_error(list(byte),Salt)`
`Salt` is not exactly 16 bytes long:
`domain_error(bcrypt_salt,Salt)`

---

password_hash/4

Computes a structured password-hash term using the selected password-hashing method and derivation options.

Compilation flags:

`static`

Template:

`password_hash(Method,Password,PasswordHash,Options)`

Mode and number of proofs:

`password_hash(+object_identifier,+list(byte),-compound,+list(compound))` - `one_or_error`

`password_hash(+compound,+list(byte),-compound,+list(compound))` - `one_or_error`

Exceptions:

`Method` is not a ground term:

`instantiation_error`

`Method` is a ground `Hash` or `pbkdf2(Hash)` term but not an object implementing the `hash_digest_protocol` protocol:

`domain_error(crypto_hash,Hash)`

`Password` is a partial list or a list with an element which is a variable:

`instantiation_error`

`Password` is a list but not a list of bytes:

`type_error(list(byte),Password)`

`Password` contains a non-integer element:

`type_error(integer,Byte)`

`Password` contains an integer outside the byte range:

`domain_error(byte,Byte)`

`Password` contains more than 72 bytes when using bcrypt:

`domain_error(bcrypt_password_length,Password)`

`Options` is a partial list or a list with an element which is a variable:

`instantiation_error`

`Options` is a list but not a list of compound terms:

`type_error(list(compound),Options)`

`Options` contains an invalid option term:

`domain_error(password_hash_option,Option)`

`Options` contains an `iterations/1` value that is not an integer:

`type_error(integer,Iterations)`

`Options` contains an `iterations/1` value that is not a positive integer:

`domain_error(positive_integer,Iterations)`

`Options` contains a `cost/1` value that is not an integer:

`type_error(integer,Cost)`

`Options` contains a `cost/1` value outside the range 4 through 31:

`domain_error(bcrypt_cost,Cost)`

`Options` contains a `salt/1` value that is not a list of bytes:

`type_error(list(byte),Salt)`

`Options` contains a `salt/1` value with a variable byte:

`instantiation_error`

`Options` contains a `salt/1` value with a non-integer byte:

`type_error(integer,Byte)`

`Options` contains a `salt/1` value with an integer outside the byte range:
`domain_error(byte,Byte)`
`Options` contains a bcrypt `salt/1` value that is not exactly 16 bytes long:
`domain_error(bcrypt_salt,Salt)`
`Options` contains an invalid APR1 `salt/1` value:
`domain_error(apr1_salt,Salt)`
`Options` contains a `salt_length/1` or `length/1` value that is not an integer:
`type_error(integer,Length)`
`Options` contains a PBKDF2 `salt_length/1` value that is less than zero:
`domain_error(non_negative_integer,Length)`
`Options` contains an APR1 `salt_length/1` value outside the range 1 through 8:
`domain_error(apr1_salt_length,Length)`
`Options` contains a `length/1` value that is not positive:
`domain_error(positive_integer,Length)`
`Options` selects a PBKDF2 `length/1` value that exceeds the maximum output length:
`domain_error(pbkdf2_output_length,Length)`

Remarks:

- Method: Supported methods are `pbkdf2(Hash)`, `bcrypt`, and `apr1`. A hash object is interpreted as shorthand for `pbkdf2(Hash)`.
- Repeated options: When the same password-hash option is given multiple times, the last occurrence is used.
- Option `iterations(Count)`: Uses the given positive integer PBKDF2 iteration count. When this option is absent, the iteration count defaults to `131072`.
- Option `cost(Count)`: Uses the given bcrypt cost from 4 through 31. When this option is absent, the cost defaults to `12`.
- Option `salt(Bytes)`: Uses the given method-specific salt. Bcrypt requires exactly 16 raw bytes; APR1 requires one through eight characters from its base64 alphabet.
- Option `salt_length(Count)`: Generates a random PBKDF2 or APR1 salt when `salt/1` is absent. PBKDF2 defaults to 16 bytes and APR1 defaults to 8 characters.
- Option `length(Count)`: Uses the given positive PBKDF2 derived-key length. When this option is absent, the derived-key length defaults to the selected hash digest size.

---

password_hash_needs_rehash/3

Succeeds when the given password-hash term does not match the selected password-hashing method and policy.

Compilation flags:
`static`

Template:
`password_hash_needs_rehash(PasswordHash,Method,Options)`
Mode and number of proofs:
`password_hash_needs_rehash(+compound,+object_identifier,+list(compound))` - `zero_or_one_or_error`
`password_hash_needs_rehash(+compound,+compound,+list(compound))` - `zero_or_one_or_error`

Exceptions:
`PasswordHash` is not a supported password-hash term:

`domain_error(password_hash,PasswordHash)`

`PasswordHash` contains a variable hash object:

`instantiation_error`

`PasswordHash` contains a hash object that does not implement the `hash_digest_protocol` protocol:

`domain_error(crypto_hash,Hash)`

`PasswordHash` contains an `Iterations` value that is not an integer:

`type_error(integer,Iterations)`

`PasswordHash` contains an `Iterations` value that is not a positive integer:

`domain_error(positive_integer,Iterations)`

`PasswordHash` contains a `Salt`, `StoredKey`, `StoredDigest`, or `Checksum` value that is not a list of bytes:

`type_error(list(byte),Bytes)`

`PasswordHash` contains a `Salt`, `StoredKey`, `StoredDigest`, or `Checksum` value with a variable byte:

`instantiation_error`

`PasswordHash` contains a `Salt`, `StoredKey`, `StoredDigest`, or `Checksum` value with a non-integer byte:

`type_error(integer,Byte)`

`PasswordHash` contains a `Salt`, `StoredKey`, `StoredDigest`, or `Checksum` value with an integer outside the byte range:

`domain_error(byte,Byte)`

`PasswordHash` contains an invalid APR1 salt:

`domain_error(apr1_salt,Salt)`

`PasswordHash` contains an invalid APR1 checksum:

`domain_error(apr1_checksum,Checksum)`

`PasswordHash` contains a bcrypt cost that is not an integer:

`type_error(integer,Cost)`

`PasswordHash` contains a bcrypt cost outside the range 4 through 31:

`domain_error(bcrypt_cost,Cost)`

`PasswordHash` contains a bcrypt salt that is not exactly 16 bytes long:

`domain_error(bcrypt_salt,Salt)`

`PasswordHash` contains an invalid bcrypt checksum:

`domain_error(bcrypt_checksum,Checksum)`

`Method` is not a ground term:

`instantiation_error`

`Method` is a ground `Hash` or `pbkdf2(Hash)` term but not an object implementing the `hash_digest_protocol` protocol:

`domain_error(crypto_hash,Hash)`

`Options` is a partial list or a list with an element which is a variable:

`instantiation_error`

`Options` is a list but not a list of compound terms:

`type_error(list(compound),Options)`

`Options` contains an invalid option term:

`domain_error(password_hash_option,Option)`

`Options` contains an `iterations/1` value that is not an integer:

`type_error(integer,Iterations)`

`Options` contains an `iterations/1` value that is not a positive integer:

`domain_error(positive_integer,Iterations)`

`Options` contains a `cost/1` value that is not an integer:

`type_error(integer,Cost)`

`Options` contains a `cost/1` value outside the range 4 through 31:

`domain_error(bcrypt_cost,Cost)`

`Options` contains a `salt/1` value that is not a list of bytes:
`type_error(list(byte),Salt)`
`Options` contains a `salt/1` value with a variable byte:
`instantiation_error`
`Options` contains a `salt/1` value with a non-integer byte:
`type_error(integer,Byte)`
`Options` contains a `salt/1` value with an integer outside the byte range:
`domain_error(byte,Byte)`
`Options` contains a bcrypt `salt/1` value that is not exactly 16 bytes long:
`domain_error(bcrypt_salt,Salt)`
`Options` contains an invalid APR1 `salt/1` value:
`domain_error(apr1_salt,Salt)`
`Options` contains a `salt_length/1` or `length/1` value that is not an integer:
`type_error(integer,Length)`
`Options` contains a PBKDF2 `salt_length/1` value that is less than zero:
`domain_error(non_negative_integer,Length)`
`Options` contains an APR1 `salt_length/1` value outside the range 1 through 8:
`domain_error(apr1_salt_length,Length)`
`Options` contains a `length/1` value that is not positive:
`domain_error(positive_integer,Length)`
`Options` selects a PBKDF2 `length/1` value that exceeds the maximum output length:
`domain_error(pbkdf2_output_length,Length)`

Remarks:

- Method: Supported target methods are `pbkdf2(Hash)`, `bcrypt`, and `apr1`. A hash object is interpreted as shorthand for `pbkdf2(Hash)`.
- Policy: The `Method` and `Options` arguments use the same method and policy options as `password_hash/4`.
- Method changes: A valid stored hash using a different method always needs rehashing.

`verify_password_hash/2`

Succeeds when the password byte sequence matches the given structured password-hash term or stored digest term.

Compilation flags:
`static`

Template:
`verify_password_hash(PasswordHash,Password)`
Mode and number of proofs:
`verify_password_hash(+compound,+list(byte))` - `zero_or_one_or_error`

Exceptions:
`PasswordHash` is not a ground term:
`instantiation_error`
`PasswordHash` is not a supported password-hash term:
`domain_error(password_hash,PasswordHash)`
`PasswordHash` contains a variable hash object:

instantiation_error

PasswordHash contains a hash object that does not implement the hash_digest_protocol protocol:

domain_error(crypto_hash,Hash)

PasswordHash contains an Iterations value that is not an integer:

type_error(integer,Iterations)

PasswordHash contains an Iterations value that is not a positive integer:

domain_error(positive_integer,Iterations)

PasswordHash contains a Salt, StoredKey, StoredDigest, or Checksum value that is not a list of bytes:

type_error(list(byte),Bytes)

PasswordHash contains a Salt, StoredKey, StoredDigest, or Checksum value with a variable byte:

instantiation_error

PasswordHash contains a Salt, StoredKey, StoredDigest, or Checksum value with a non-integer byte:

type_error(integer,Byte)

PasswordHash contains a Salt, StoredKey, StoredDigest, or Checksum value with an integer outside the byte range:

domain_error(byte,Byte)

PasswordHash contains an invalid APR1 salt:

domain_error(apr1_salt,Salt)

PasswordHash contains an invalid APR1 checksum:

domain_error(apr1_checksum,Checksum)

Password is a partial list or a list with an element which is a variable:

instantiation_error

Password is a list but not a list of bytes:

type_error(list(byte),Password)

Password contains a non-integer element:

type_error(integer,Byte)

Password contains an integer outside the byte range:

domain_error(byte,Byte)

---

password_hash_atom/2

Converts between a structured bcrypt 2b or APR1 password-hash term and its canonical modular-crypt atom.

Compilation flags:

static

Template:

password_hash_atom(PasswordHash,Atom)

Mode and number of proofs:

password_hash_atom(+compound,?atom) - zero_or_one_or_error

password_hash_atom(-compound,+atom) - one_or_error

Exceptions:

Both arguments are variables:

instantiation_error

Atom is neither a variable nor an atom:

type_error(atom,Atom)

The instantiated argument does not represent a supported canonical password hash:
`domain_error(password_hash,Value)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

xor_bytes/3

Computes the byte-wise exclusive disjunction of two equal-length byte lists.

Compilation flags:
`static`

Template:
`xor_bytes(Bytes1,Bytes2,XorBytes)`
Mode and number of proofs:
`xor_bytes(+list(byte),+list(byte),-list(byte))` - `one`

---

constant_time_equal/2

Compares two byte lists using a constant-time strategy.

Compilation flags:
`static`

Template:
`constant_time_equal(Expected,Candidate)`
Mode and number of proofs:
`constant_time_equal(+list(byte),+list(byte))` - `zero_or_one`

Operators

(none)
category

1.55.3 ed25519

Ed25519 (RFC 8032) public-key signature implementation. Requires exact, unbounded integer arithmetic for arithmetic modulo the 255-bit field prime and modulo the group order.

Availability:
`logtalk_load(crypto(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-21

Compilation flags:
`static`

Complements:
crypto
Uses:
list
sha512
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *ed25519_keypair/2*
  - *ed25519_public_key/2*
  - *ed25519_sign/3*
  - *ed25519_verify/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

ed25519_keypair/2

Generates a fresh random 32-byte Seed and derives the corresponding 32-byte PublicKey. The seed is the value to keep secret and to pass to ed25519_sign/3; it is also sometimes called the "secret key". Available only on backends with unbounded integer arithmetic.

Compilation flags:
`static`

Template:

```
ed25519_keypair(Seed,PublicKey)
```

Mode and number of proofs:

```
ed25519_keypair(-list(byte),-list(byte)) - one
```

---

ed25519_public_key/2

Derives the 32-byte PublicKey corresponding to a 32-byte Seed. Available only on backends with unbounded integer arithmetic.

Compilation flags:

```
static
```

Template:

```
ed25519_public_key(Seed,PublicKey)
```

Mode and number of proofs:

```
ed25519_public_key(+list(byte),-list(byte)) - one_or_error
```

Exceptions:

`Seed` is a partial list or a list with an element which is a variable:

```
instantiation_error
```

`Seed` is neither a variable nor a list of 32 bytes:

```
type_error(list(byte,32),Seed)
```

`Seed` contains a non-integer byte:

```
type_error(integer,Byte)
```

`Seed` contains an integer outside the byte range:

```
domain_error(byte,Byte)
```

---

ed25519_sign/3

Computes the 64-byte Ed25519 Signature of Message using Seed. Not constant-time; see the module-level note on this section. Available only on backends with unbounded integer arithmetic.

Compilation flags:

```
static
```

Template:

```
ed25519_sign(Seed,Message,Signature)
```

Mode and number of proofs:

```
ed25519_sign(+list(byte),+list(byte),-list(byte)) - one_or_error
```

Exceptions:

`Seed` is a partial list or a list with an element which is a variable:

```
instantiation_error
```

`Seed` is neither a variable nor a list of 32 bytes:

```
type_error(list(byte,32),Seed)
```

`Seed` contains a non-integer byte:

`type_error(integer,Byte)`
`Seed` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Message` is a partial list or a list with an element which is a variable:
`instantiation_error`
`Message` is neither a variable nor a list of bytes:
`type_error(list(byte),Message)`
`Message` contains a non-integer byte:
`type_error(integer,Byte)`
`Message` contains an integer outside the byte range:
`domain_error(byte,Byte)`

`ed25519_verify/3`

Succeeds if Signature is a valid Ed25519 signature of Message under PublicKey, and fails otherwise (invalid signature, non-canonical S, non-canonical or small-order PublicKey or R component, or malformed input lengths); see the module-level note on this section for the exact verification policy. Available only on backends with unbounded integer arithmetic.

Compilation flags:
`static`

Template:
`ed25519_verify(PublicKey,Message,Signature)`
Mode and number of proofs:
`ed25519_verify(+list(byte),+list(byte),+list(byte))` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.55.4 x25519

X25519 Diffie-Hellman key agreement implementation (RFC 7748). Requires exact, unbounded integer arithmetic for arithmetic modulo the 255-bit field prime.

Availability:
`logtalk_load(crypto(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-25

Compilation flags:
`static`

Complements:
crypto
Uses:
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *x25519_keypair/2*
  - *x25519_public_key/2*
  - *x25519_shared_secret/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

x25519_keypair/2

Generates a fresh random 32-byte PrivateKey and derives the corresponding 32-byte PublicKey. Available only on backends with unbounded integer arithmetic.

Compilation flags:
`static`

Template:
`x25519_keypair(PrivateKey,PublicKey)`
Mode and number of proofs:

```
x25519_keypair(-list(byte),-list(byte)) - one
```

---

x25519_public_key/2

Derives the 32-byte X25519 PublicKey corresponding to a 32-byte PrivateKey. Available only on backends with unbounded integer arithmetic.

Compilation flags:

```
static
```

Template:

```
x25519_public_key(PrivateKey,PublicKey)
```

Mode and number of proofs:

```
x25519_public_key(+list(byte),-list(byte)) - one_or_error
```

Exceptions:

`PrivateKey` is a partial list or a list with an element which is a variable:

```
instantiation_error
```

`PrivateKey` is neither a variable nor a list of 32 bytes:

```
type_error(list(byte,32),PrivateKey)
```

`PrivateKey` contains a non-integer byte:

```
type_error(integer,Byte)
```

`PrivateKey` contains an integer outside the byte range:

```
domain_error(byte,Byte)
```

---

x25519_shared_secret/3

Computes the 32-byte X25519 SharedSecret using PrivateKey and PeerPublicKey. Fails when the result is all zeroes. Available only on backends with unbounded integer arithmetic.

Compilation flags:

```
static
```

Template:

```
x25519_shared_secret(PrivateKey,PeerPublicKey,SharedSecret)
```

Mode and number of proofs:

```
x25519_shared_secret(+list(byte),+list(byte),-list(byte)) - zero_or_one_or_error
```

Exceptions:

`PrivateKey` is a partial list or a list with an element which is a variable:

```
instantiation_error
```

`PrivateKey` is neither a variable nor a list of 32 bytes:

```
type_error(list(byte,32),PrivateKey)
```

`PrivateKey` contains a non-integer byte:

```
type_error(integer,Byte)
```

`PrivateKey` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`PeerPublicKey` is a partial list or a list with an element which is a variable:
`instantiation_error`
`PeerPublicKey` is neither a variable nor a list of 32 bytes:
`type_error(list(byte,32),PeerPublicKey)`
`PeerPublicKey` contains a non-integer byte:
`type_error(integer,Byte)`
`PeerPublicKey` contains an integer outside the byte range:
`domain_error(byte,Byte)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.55.5 `xchacha20_poly1305`

XChaCha20 stream cipher and XChaCha20-Poly1305 authenticated encryption with associated data algorithm implementation. Requires exact, unbounded integer arithmetic for the Poly1305 130-bit accumulator.

Availability:
`logtalk_load(crypto(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-21

Compilation flags:
`static`

Complements:
crypto
Uses:
list
type

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *xchacha20/4*
  - *xchacha20_subkey_and_nonce/4*
  - *chacha20_encrypt/5*
  - *chacha20_block/4*
  - *xchacha20_poly1305_encrypt/5*
  - *xchacha20_poly1305_decrypt/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

xchacha20/4

Encrypts or decrypts Input using XChaCha20. Clients should only use this unauthenticated stream cipher as part of a construction that provides its own authentication.

Compilation flags:

```
static
```

Template:

```
xchacha20(Key,Nonce,Input,Output)
```

Mode and number of proofs:

```
xchacha20(+list(byte),+list(byte),+list(byte),-list(byte)) - one_or_error
```

Exceptions:

`Key` is a partial list or a list with an element which is a variable:

```
instantiation_error
```

`Key` is neither a variable nor a list of 32 bytes:

```
type_error(list(byte,32),Key)
```

`Key` contains a non-integer byte:

```
type_error(integer,Byte)
```

`Key` contains an integer outside the byte range:

```
domain_error(byte,Byte)
```

`Nonce` is a partial list or a list with an element which is a variable:

```
instantiation_error
```

`Nonce` is neither a variable nor a list of 24 bytes:

```
type_error(list(byte,24),Nonce)
```

`Nonce` contains a non-integer byte:

```
type_error(integer,Byte)
```

`Nonce` contains an integer outside the byte range:

```
domain_error(byte,Byte)
```

`Input` is a partial list or a list with an element which is a variable:

```
instantiation_error
```

`Input` is neither a variable nor a list of bytes:

`type_error(list(byte),Input)`
`Input` contains a non-integer byte:
`type_error(integer,Byte)`
`Input` contains an integer outside the byte range:
`domain_error(byte,Byte)`

---

xchacha20_subkey_and_nonce/4

Derives the ChaCha20 subkey words and nonce words for an XChaCha20 key and nonce. The caller is responsible for validating the key and nonce and should only use the result in a construction that provides authentication.

Compilation flags:
`static`

Template:
`xchacha20_subkey_and_nonce(Key,Nonce,SubkeyWords,NonceWords)`
Mode and number of proofs:
`xchacha20_subkey_and_nonce(+list(byte),+list(byte),-list(integer),-list(integer))` - `one`

---

chacha20_encrypt/5

Encrypts or decrypts bytes using ChaCha20 words and the given initial block counter. The caller is responsible for validating all arguments and should only use this unauthenticated stream cipher as part of a construction that provides authentication.

Compilation flags:
`static`

Template:
`chacha20_encrypt(KeyWords,Counter,NonceWords,Input,Output)`
Mode and number of proofs:
`chacha20_encrypt(+list(integer),+non_negative_integer,+list(integer),+list(byte),-list(byte))` - `one`

---

chacha20_block/4

Computes a 64-byte ChaCha20 block using key words and the given block counter and nonce words. The caller is responsible for validating all arguments and should only use it as part of a construction that provides authentication.

Compilation flags:
`static`

Template:

`chacha20_block(KeyWords,Counter,NonceWords,Block)`

Mode and number of proofs:

`chacha20_block(+list(integer),+non_negative_integer,+list(integer),-list(byte))` - `one`

---

xchacha20_poly1305_encrypt/5

Encrypts Plaintext with XChaCha20 and appends a 16-byte Poly1305 authentication tag covering AAD and the ciphertext, following the IETF ChaCha20-Poly1305 AEAD construction (RFC 8439) extended with the 24-byte XChaCha nonce. Available only on backends with unbounded integer arithmetic.

Compilation flags:

`static`

Template:

`xchacha20_poly1305_encrypt(Key,Nonce,AAD,Plaintext,CiphertextAndTag)`

Mode and number of proofs:

`xchacha20_poly1305_encrypt(+list(byte),+list(byte),+list(byte),+list(byte),-list(byte))` - `one_or_error`

Exceptions:

`Key` is a partial list or a list with an element which is a variable:

`instantiation_error`

`Key` is neither a variable nor a list of 32 bytes:

`type_error(list(byte,32),Key)`

`Key` contains a non-integer byte:

`type_error(integer,Byte)`

`Key` contains an integer outside the byte range:

`domain_error(byte,Byte)`

`Nonce` is a partial list or a list with an element which is a variable:

`instantiation_error`

`Nonce` is neither a variable nor a list of 24 bytes:

`type_error(list(byte,24),Nonce)`

`Nonce` contains a non-integer byte:

`type_error(integer,Byte)`

`Nonce` contains an integer outside the byte range:

`domain_error(byte,Byte)`

`AAD` is a partial list or a list with an element which is a variable:

`instantiation_error`

`AAD` is neither a variable nor a list of bytes:

`type_error(list(byte),AAD)`

`AAD` contains a non-integer byte:

`type_error(integer,Byte)`

`AAD` contains an integer outside the byte range:

`domain_error(byte,Byte)`

`Plaintext` is a partial list or a list with an element which is a variable:

`instantiation_error`

`Plaintext` is neither a variable nor a list of bytes:

`type_error(list(byte),Plaintext)`
`Plaintext` contains a non-integer byte:
`type_error(integer,Byte)`
`Plaintext` contains an integer outside the byte range:
`domain_error(byte,Byte)`

`xchacha20_poly1305_decrypt/5`

Verifies the trailing 16-byte Poly1305 tag of CiphertextAndTag against AAD using constant-time comparison and, only if it matches, decrypts the ciphertext with XChaCha20. Fails, without decrypting anything, if the tag does not match. Available only on backends with unbounded integer arithmetic.

Compilation flags:
`static`

Template:
`xchacha20_poly1305_decrypt(Key,Nonce,AAD,CiphertextAndTag,Plaintext)`
Mode and number of proofs:
`xchacha20_poly1305_decrypt(+list(byte),+list(byte),+list(byte),+list(byte),-list(byte))` - `zero_or_one_or_error`

Exceptions:
`Key` is a partial list or a list with an element which is a variable:
`instantiation_error`
`Key` is neither a variable nor a list of 32 bytes:
`type_error(list(byte,32),Key)`
`Key` contains a non-integer byte:
`type_error(integer,Byte)`
`Key` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Nonce` is a partial list or a list with an element which is a variable:
`instantiation_error`
`Nonce` is neither a variable nor a list of 24 bytes:
`type_error(list(byte,24),Nonce)`
`Nonce` contains a non-integer byte:
`type_error(integer,Byte)`
`Nonce` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`AAD` is a partial list or a list with an element which is a variable:
`instantiation_error`
`AAD` is neither a variable nor a list of bytes:
`type_error(list(byte),AAD)`
`AAD` contains a non-integer byte:
`type_error(integer,Byte)`
`AAD` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`CiphertextAndTag` is a partial list or a list with an element which is a variable:

`instantiation_error`
`CiphertextAndTag` is neither a variable nor a list of bytes:
`type_error(list(byte),CiphertextAndTag)`
`CiphertextAndTag` contains a non-integer byte:
`type_error(integer,Byte)`
`CiphertextAndTag` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`CiphertextAndTag` contains fewer than 16 bytes:
`domain_error(minimum_byte_length(16),CiphertextAndTag)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.56 csv

object

1.56.1 `csv`

CSV files reading and writing predicates using the options Header - `keep`, Separator - `comma`, and IgnoreQuotes - `false`.

Availability:
`logtalk_load(csv(loader))`

Author: Jacinto Dávila
Version: 1:0:0
Date: 2021-02-02

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *csv(keep,comma,false,false)*

Remarks:
(none)

Inherited public predicates:

guess_arity/2 *guess_separator/2* *read_file/2* *read_file/3* *read_file_by_line/2* *read_file_by_line/3* *read_stream/2* *read_stream/3* *read_stream_by_line/2* *read_stream_by_line/3* *write_file/3* *write_stream/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.56.2 `csv(Header,Separator,IgnoreQuotes)`

- `Header` - Header handling option with possible values `missing`, `skip`, and `keep` (default).
- `Separator` - Separator handling option with possible values `comma` (default for non `.tsv` and non `.tab` files or when no file name extension is available), `tab` (default for `.tsv` and `.tab` files), `semicolon`, and `colon`.
- `IgnoreQuotes` - Double-quotes handling option to ignore (`true`) or preserve (`false`; default) double quotes surrounding data.

Backward-compatible parametric object equivalent to using `csv(_Header_, _Separator_, _IgnoreQuotes_, false)`.

Availability:
`logtalk_load(csv(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-02-27

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *csv(Header,Separator,IgnoreQuotes,false)*

Remarks:

(none)

Inherited public predicates:

guess_arity/2 guess_separator/2 read_file/2 read_file/3 read_file_by_line/2 read_file_by_line/3 read_stream/2 read_stream/3 read_stream_by_line/2 read_stream_by_line/3 write_file/3 write_stream/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.56.3 `csv(Header,Separator,IgnoreQuotes,Comments)`

- `Header` - Header handling option with possible values `missing`, `skip`, and `keep` (default).
- `Separator` - Separator handling option with possible values `comma` (default for non `.tsv` and non `.tab` files or when no file name extension is available), `tab` (default for `.tsv` and `.tab` files), `semicolon`, and `colon`.
- `IgnoreQuotes` - Double-quotes handling option to ignore (`true`) or preserve (`false`; default) double quotes surrounding data.
- `Comments` - Comment handling option with possible values `true` and `false` (default). When `true`, lines starting with `#` are skipped when reading CSV files and streams.

CSV file and stream reading and writing predicates.

Availability:

```
logtalk_load(csv(loader))
```

Author: Jacinto Dávila and Paulo Moura
Version: 2:2:0
Date: 2026-02-25

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *csv_protocol*

Uses:

list
logtalk
os
reader
type

Remarks:

(none)

Inherited public predicates:

guess_arity/2 *guess_separator/2* *read_file/2* *read_file/3* *read_file_by_line/2* *read_file_by_line/3* *read_stream/2* *read_stream/3* *read_stream_by_line/2* *read_stream_by_line/3* *write_file/3* *write_stream/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

category

1.56.4 `csv_guess_questions`

Support for asking questions when guessing the separator and the record arity of CSV files.

Availability:

`logtalk_load(csv(loader))`

Author: Jacinto Dávila
Version: 1:0:0
Date: 2021-02-03

Compilation flags:

`static`

Provides:
logtalk::message_tokens//2
logtalk::question_prompt_stream/4

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.56.5 `csv_protocol`

CSV file and stream reading and writing protocol.

Availability:
`logtalk_load(csv(loader))`

Author: Jacinto Dávila and Paulo Moura
Version: 2:0:1
Date: 2025-05-07

Compilation flags:
`static`

Dependencies:

(none)

Remarks:

- Type-checking: Some of the predicate file and stream argument type-checking exceptions depend on the Prolog backend compliance with standards.

Inherited public predicates:
(none)

- *Public predicates*
  - *read_file/3*
  - *read_stream/3*
  - *read_file/2*
  - *read_stream/2*
  - *read_file_by_line/3*
  - *read_stream_by_line/3*
  - *read_file_by_line/2*
  - *read_stream_by_line/2*
  - *write_file/3*
  - *write_stream/3*
  - *guess_separator/2*
  - *guess_arity/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

read_file/3

Reads a CSV file saving the data as clauses for the specified object predicate. Fails if the file cannot be parsed.

Compilation flags:
`static`

Template:
`read_file(File,Object,Predicate)`

Mode and number of proofs:
`read_file(+atom,+object_identifier,+predicate_indicator)` - `zero_or_one`

Exceptions:
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`File` is an atom but not an existing file:
`existence_error(file,File)`

`File` is an existing file but cannot be opened for reading:
`permission_error(open,source_sink,File)`
`Object` is a variable:
`instantiation_error`
`Object` is neither a variable nor an object identifier:
`type_error(object_identifier,Object)`
`Object` is a valid object identifier but not an existing object:
`existence_error(object,Object)`
`Predicate` is a variable:
`instantiation_error`
`Predicate` is neither a variable nor a predicate indicator:
`type_error(predicate_indicator,Predicate)`
`Predicate` is a valid predicate indicator but not an existing public predicate:
`existence_error(predicate,Predicate)`

`read_stream/3`

Reads a CSV stream saving the data as clauses for the specified object predicate. Fails if the stream cannot be parsed.

Compilation flags:
`static`

Template:
`read_stream(Stream,Object,Predicate)`
Mode and number of proofs:
`read_stream(+stream_or_alias,+object_identifier,+predicate_indicator)` - `zero_or_one`

Exceptions:
`Stream` is a variable:
`instantiation_error`
`Stream` is neither a variable nor a stream-term or alias:
`domain_error(stream_or_alias,Stream)`
`Stream` is not an open stream:
`existence_error(stream,Stream)`
`Stream` is an output stream:
`permission_error(input,stream,Stream)`
`Stream` is a binary stream:
`permission_error(input,binary_stream,Stream)`
`Object` is a variable:
`instantiation_error`
`Object` is neither a variable nor an object identifier:
`type_error(object_identifier,Object)`
`Object` is a valid object identifier but not an existing object:
`existence_error(object,Object)`
`Predicate` is a variable:
`instantiation_error`

`Predicate` is neither a variable nor a predicate indicator:
`type_error(predicate_indicator,Predicate)`
`Predicate` is a valid predicate indicator but not an existing public predicate:
`existence_error(predicate,Predicate)`

---

read_file/2

Reads a CSV file returning the data as a list of rows, each row a list of fields. Fails if the file cannot be parsed.

Compilation flags:
`static`

Template:
`read_file(File,Rows)`
Mode and number of proofs:
`read_file(+atom,-list(list))` - `zero_or_one`

Exceptions:
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`File` is an atom but not an existing file:
`existence_error(file,File)`
`File` is an existing file but cannot be opened for reading:
`permission_error(open,source_sink,File)`

---

read_stream/2

Reads a CSV stream returning the data as a list of rows, each row a list of fields. Fails if the stream cannot be parsed.

Compilation flags:
`static`

Template:
`read_stream(Stream,Rows)`
Mode and number of proofs:
`read_stream(+stream_or_alias,-list(list))` - `zero_or_one`

Exceptions:
`Stream` is a variable:
`instantiation_error`
`Stream` is neither a variable nor a stream-term or alias:
`domain_error(stream_or_alias,Stream)`

`Stream` is not an open stream:
`existence_error(stream,Stream)`
`Stream` is an output stream:
`permission_error(input,stream,Stream)`
`Stream` is a binary stream:
`permission_error(input,binary_stream,Stream)`

`read_file_by_line/3`

Reads a CSV file saving the data as clauses for the specified object predicate. The file is read line by line. Fails if the file cannot be parsed.

Compilation flags:
`static`

Template:
`read_file_by_line(File,Object,Predicate)`
Mode and number of proofs:
`read_file_by_line(+atom,+object_identifier,+predicate_indicator)` - `zero_or_one`

Exceptions:
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`File` is an atom but not an existing file:
`existence_error(file,File)`
`File` is an existing file but cannot be opened for reading:
`permission_error(open,source_sink,File)`
`Object` is a variable:
`instantiation_error`
`Object` is neither a variable nor an object identifier:
`type_error(object_identifier,Object)`
`Object` is a valid object identifier but not an existing object:
`existence_error(object,Object)`
`Predicate` is a variable:
`instantiation_error`
`Predicate` is neither a variable nor a predicate indicator:
`type_error(predicate_indicator,Predicate)`
`Predicate` is a valid predicate indicator but not an existing public predicate:
`existence_error(predicate,Predicate)`

---

read_stream_by_line/3

Reads a CSV stream saving the data as clauses for the specified object predicate. The stream is read line by line. Fails if the stream cannot be parsed.

Compilation flags:

`static`

Template:

`read_stream_by_line(Stream,Object,Predicate)`

Mode and number of proofs:

`read_stream_by_line(+stream_or_alias,+object_identifier,+predicate_indicator)` - `zero_or_one`

Exceptions:

`Stream` is a variable:

`instantiation_error`

`Stream` is neither a variable nor a stream-term or alias:

`domain_error(stream_or_alias,Stream)`

`Stream` is not an open stream:

`existence_error(stream,Stream)`

`Stream` is an output stream:

`permission_error(input,stream,Stream)`

`Stream` is a binary stream:

`permission_error(input,binary_stream,Stream)`

`Object` is a variable:

`instantiation_error`

`Object` is neither a variable nor an object identifier:

`type_error(object_identifier,Object)`

`Object` is a valid object identifier but not an existing object:

`existence_error(object,Object)`

`Predicate` is a variable:

`instantiation_error`

`Predicate` is neither a variable nor a predicate indicator:

`type_error(predicate_indicator,Predicate)`

`Predicate` is a valid predicate indicator but not an existing public predicate:

`existence_error(predicate,Predicate)`

---

read_file_by_line/2

Reads a CSV file returning the data as a list of rows, each row a list of fields. The file is read line by line. Fails if the file cannot be parsed.

Compilation flags:

`static`

Template:

`read_file_by_line(File,Rows)`

Mode and number of proofs:

`read_file_by_line(+atom,-list(list))` - `zero_or_one`

Exceptions:

`File` is a variable:

`instantiation_error`

`File` is neither a variable nor an atom:

`type_error(atom,File)`

`File` is an atom but not an existing file:

`existence_error(file,File)`

`File` is an existing file but cannot be opened for reading:

`permission_error(open,source_sink,File)`

---

read_stream_by_line/2

Reads a CSV stream returning the data as a list of rows, each row a list of fields. The stream is read line by line. Fails if the stream cannot be parsed.

Compilation flags:

`static`

Template:

`read_stream_by_line(Stream,Rows)`

Mode and number of proofs:

`read_stream_by_line(+stream_or_alias,-list(list))` - `zero_or_one`

Exceptions:

`Stream` is a variable:

`instantiation_error`

`Stream` is neither a variable nor a stream-term or alias:

`domain_error(stream_or_alias,Stream)`

`Stream` is not an open stream:

`existence_error(stream,Stream)`

`Stream` is an output stream:

`permission_error(input,stream,Stream)`

`Stream` is a binary stream:

`permission_error(input,binary_stream,Stream)`

---

write_file/3

Writes a CSV file with the data represented by the solutions to the specified object predicate.

Compilation flags:
`static`

Template:
`write_file(File,Object,Predicate)`
Mode and number of proofs:
`write_file(+atom,+object_identifier,+predicate_indicator)` - `one`

Exceptions:
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`File` is an atom but cannot be opened for writing:
`permission_error(open,source_sink,File)`
`Object` is a variable:
`instantiation_error`
`Object` is neither a variable nor an object identifier:
`type_error(object_identifier,Object)`
`Object` is a valid object identifier but not an existing object:
`existence_error(object,Object)`
`Predicate` is a variable:
`instantiation_error`
`Predicate` is neither a variable nor a predicate indicator:
`type_error(predicate_indicator,Predicate)`
`Predicate` is a valid predicate indicator but not an existing public predicate:
`existence_error(predicate,Predicate)`

---

write_stream/3

Writes a CSV stream with the data represented by the solutions to the specified object predicate.

Compilation flags:
`static`

Template:
`write_stream(Stream,Object,Predicate)`
Mode and number of proofs:
`write_stream(+stream_or_alias,+object_identifier,+predicate_indicator)` - `one`

Exceptions:
`Stream` is a variable:
`instantiation_error`

`Stream` is neither a variable nor a stream-term or alias:
`domain_error(stream_or_alias,Stream)`
`Stream` is not an open stream:
`existence_error(stream,Stream)`
`Stream` is an input stream:
`permission_error(output,stream,Stream)`
`Stream` is a binary stream:
`permission_error(output,binary_stream,Stream)`
`Object` is a variable:
`instantiation_error`
`Object` is neither a variable nor an object identifier:
`type_error(object_identifier,Object)`
`Object` is a valid object identifier but not an existing object:
`existence_error(object,Object)`
`Predicate` is a variable:
`instantiation_error`
`Predicate` is neither a variable nor a predicate indicator:
`type_error(predicate_indicator,Predicate)`
`Predicate` is a valid predicate indicator but not an existing public predicate:
`existence_error(predicate,Predicate)`

---

guess_separator/2

Guesses the separator used in a given file, asking the user to confirm.

Compilation flags:
`static`

Template:
`guess_separator(File,Separator)`
Mode and number of proofs:
`guess_separator(+atom,-atom)` - `one`

Exceptions:
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`File` is an atom but not an existing file:
`existence_error(file,File)`
`File` is an atom but cannot be opened for writing:
`permission_error(open,source_sink,File)`

---

guess_arity/2

Guesses the arity of records in a given file, asking the user to confirm.

Compilation flags:
`static`

Template:
`guess_arity(File,Arity)`
Mode and number of proofs:
`guess_arity(+atom,-number)` - `one`

Exceptions:
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`File` is an atom but not an existing file:
`existence_error(file,File)`
`File` is an atom but cannot be opened for writing:
`permission_error(open,source_sink,File)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.57 cuid2

object

1.57.1 cuid2

Cuid2 generator using atom representation, 24 symbols, and a lowercase alphanumeric alphabet.

Availability:
`logtalk_load(cuid2(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *cuid2(atom,24,abcdefghijklmnopqrstuvwxyz0123456789)*

Remarks:
(none)

Inherited public predicates:
generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

cuid2(Representation,Size,Alphabet), *ids*, *ksuid*, *nanoid*, *snowflakeid*, *sqids*, *typeid*, *ulid*, *uuid*

object

1.57.2 `cuid2(Representation,Size,Alphabet)`

- `Representation` - Text representation for the Cuid2 identifier. Possible values are `atom`, `chars`, and `codes`.
- `Size` - Number of symbols in the Cuid2 identifier.
- `Alphabet` - Alphabet used for generating Cuid2 identifiers represented as an atom, list of characters, or list of character codes.

Cuid2 generator.

Availability:
`logtalk_load(cuid2(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *cuid2_protocol*
Uses:
crypto
list

Remarks:
(none)

Inherited public predicates:
generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

cuid2, *ids(Representation,Bytes)*, *ksuid(Representation,Alphabet)*, *nanoid(Representation,Size,Alphabet)*, *snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node)*, *sqids(Representation)*, *typeid(Representation)*, *ulid(Representation)*, *uuid(Representation)*

protocol

1.57.3 cuid2_protocol

Cuid2 generator protocol.

Availability:
`logtalk_load(cuid2(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-26

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *generate/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

generate/1

Returns a Cuid2 identifier.

Compilation flags:
`static`

Template:
`generate(Cuid2)`
Mode and number of proofs:
`generate(--ground)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.58 cusum_anomaly_detector

object

1.58.1 `cusum_anomaly_detector`

CUSUM (Cumulative Sum Control Chart) anomaly detector for continuous sequence-like datasets. Learns per-step population mean and standard deviation from baseline training examples, or from the baseline subset of a labeled dataset selected by learn-time class-label options, using a dataset object implementing the `anomaly_dataset_protocol` protocol. Returns a detector term that can be used for scoring, prediction, and export.

Availability:
`logtalk_load(cusum_anomaly_detector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *anomaly_detector_common*

Uses:
format
list
pairs
population
type

Remarks:
(none)

Inherited public predicates:
anomaly_detector_options/2 *check_anomaly_detector/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict/4* *print_anomaly_detector/1* *score/3* *score_all/3* *valid_anomaly_detector/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

anomaly_dataset_protocol, *anomaly_detector_protocol*, *isolation_forest_anomaly_detector*, *knn_distance_anomaly_detector*, *lof_anomaly_detector*, *modified_z_score_anomaly_detector*, *z_score_anomaly_detector*

1.59 datalog

object

1.59.1 `datalog`

Portable Datalog engine with stratified negation and incremental updates.

Availability:
`logtalk_load(datalog(loader))`

Author: Paulo Moura
Version: 0:2:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *datalog_protocol*
Uses:
list
numberlist

type
varlist

Remarks:
(none)

Inherited public predicates:
add_rule/3 *assert_fact/1* *begin/0* *clear/0* *commit/0* *explain/2* *facts/1* *load_program/1* *materialize/0* *predicate_stratum/3* *query/1* *query/2* *remove_rule/1* *retract_fact/1* *rollback/0* *rules/1* *strata/1* *update/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - `rule_/3`
  - `edb_fact_/1`
  - `idb_fact_/1`
  - `support_count_/2`
  - `support_edge_/3`
  - `predicate_stratum_/3`
  - `snapshot_/6`
  - `restore_snapshot/6`
  - `restore_edb_facts/1`
  - `restore_idb_facts/1`
  - `restore_support_counts/1`
  - `restore_support_edges/1`
  - `restore_predicate_strata/1`
  - `strata_from_numbers/2`
  - `predicates_in_stratum/2`
  - `has_aggregate_rules/0`
  - `aggregate_body_predicate/2`
  - `derive_aggregate_literal/2`
  - `derive_aggregate_goals/1`
  - `optimize_rule_body/2`
  - `partition_body_literals/5`
  - `literal_bucket/2`
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

rule_/3

Table of loaded rules represented as rule id, head, and body literals.

Compilation flags:
`dynamic`

Template:
`rule_(Id,Head,Body)`
Mode and number of proofs:
`rule_(?nonvar,?callable,?list)` - `zero_or_more`

---

edb_fact_/1

Table of extensional database (EDB) facts.

Compilation flags:
`dynamic`

Template:
`edb_fact_(Fact)`
Mode and number of proofs:
`edb_fact_(?callable)` - `zero_or_more`

---

idb_fact_/1

Table of intensional database (IDB) currently derived facts.

Compilation flags:
`dynamic`

Template:
`idb_fact_(Fact)`
Mode and number of proofs:
`idb_fact_(?callable)` - `zero_or_more`

---

support_count_/2

Table of derivation support counts for currently derived facts.

Compilation flags:
```
dynamic
```

Template:
```
support_count_(Fact,Count)
```
Mode and number of proofs:
```
support_count_(?callable,?integer) - zero_or_more
```

---

support_edge_/3

Table of concrete derivation edges as fact, rule id, and supporting literals.

Compilation flags:
```
dynamic
```

Template:
```
support_edge_(Fact,RuleId,Supports)
```
Mode and number of proofs:
```
support_edge_(?callable,?nonvar,?list) - zero_or_more
```

---

predicate_stratum_/3

Table of computed predicate strata represented as name, arity, and stratum number.

Compilation flags:
```
dynamic
```

Template:
```
predicate_stratum_(Name,Arity,Stratum)
```
Mode and number of proofs:
```
predicate_stratum_(?atom,?integer,?integer) - zero_or_more
```

---

snapshot_/6

Transaction snapshot of rules, EDB facts, IDB facts, support counts, support edges, and predicate strata.

Compilation flags:
`dynamic`

Template:
`snapshot_(Rules,EdbFacts,IdbFacts,SupportCounts,SupportEdges,PredicateStrata)`
Mode and number of proofs:
`snapshot_(?list,?list,?list,?list,?list,?list)` - `zero_or_one`

---

restore_snapshot/6

Restores a saved transaction snapshot into the current engine state.

Compilation flags:
`static`

Template:
`restore_snapshot(Rules,EdbFacts,IdbFacts,SupportCounts,SupportEdges,`
`PredicateStrata)`
Mode and number of proofs:
`restore_snapshot(+list,+list,+list,+list,+list,+list)` - `one`

---

restore_edb_facts/1

Restores EDB facts from a saved list.

Compilation flags:
`static`

Template:
`restore_edb_facts(Facts)`
Mode and number of proofs:
`restore_edb_facts(+list(callable))` - `one`

---

restore_idb_facts/1

Restores IDB facts from a saved list.

Compilation flags:
```
static
```

Template:
```
restore_idb_facts(Facts)
```
Mode and number of proofs:
```
restore_idb_facts(+list(callable)) - one
```

---

restore_support_counts/1

Restores support counts from a saved list.

Compilation flags:
```
static
```

Template:
```
restore_support_counts(Supports)
```
Mode and number of proofs:
```
restore_support_counts(+list) - one
```

---

restore_support_edges/1

Restores support edges from a saved list.

Compilation flags:
```
static
```

Template:
```
restore_support_edges(Edges)
```
Mode and number of proofs:
```
restore_support_edges(+list) - one
```

---

restore_predicate_strata/1

Restores predicate strata from a saved list.

Compilation flags:
`static`

Template:
`restore_predicate_strata(Strata)`
Mode and number of proofs:
`restore_predicate_strata(+list)` - `one`

---

strata_from_numbers/2

Builds grouped strata terms from a sorted list of stratum numbers.

Compilation flags:
`static`

Template:
`strata_from_numbers(StratumNumbers,Strata)`
Mode and number of proofs:
`strata_from_numbers(+list(integer),-list)` - `one`

---

predicates_in_stratum/2

Returns predicates in a given stratum as sorted `predicate(Name, Arity)` terms.

Compilation flags:
`static`

Template:
`predicates_in_stratum(Stratum,Predicates)`
Mode and number of proofs:
`predicates_in_stratum(+integer,-list)` - `one`

---

has_aggregate_rules/0

True when at least one loaded rule body contains an aggregate literal.

Compilation flags:
`static`

Mode and number of proofs:
`has_aggregate_rules` - `zero_or_one`

---

aggregate_body_predicate/2

Enumerates predicate indicators referenced in aggregate goals from a body literal list.

Compilation flags:
`static`

Template:
`aggregate_body_predicate(Body,Predicate)`
Mode and number of proofs:
`aggregate_body_predicate(+list,-compound)` - `zero_or_more`

---

derive_aggregate_literal/2

Evaluates an aggregate literal and returns a normalized support term.

Compilation flags:
`static`

Template:
`derive_aggregate_literal(AggregateLiteral,Support)`
Mode and number of proofs:
`derive_aggregate_literal(+compound,-nonvar)` - `zero_or_one`

---

derive_aggregate_goals/1

Succeeds when all aggregate goals are true for the current bindings.

Compilation flags:
`static`

Template:

```
derive_aggregate_goals(Goals)
```

Mode and number of proofs:

```
derive_aggregate_goals(+list(callable)) - zero_or_one
```

---

optimize_rule_body/2

Normalizes a rule body by placing positive ground literals first, then positive non-ground literals, then aggregates, and finally negative literals.

Compilation flags:

```
static
```

Template:

```
optimize_rule_body(Body,OptimizedBody)
```

Mode and number of proofs:

```
optimize_rule_body(+list,-list) - one
```

---

partition_body_literals/5

Partitions body literals into positive-ground, positive-non-ground, aggregate, and negative lists preserving relative order.

Compilation flags:

```
static
```

Template:

```
partition_body_literals(Body,PositiveGround,PositiveNonGround,Aggregates,
Negatives)
```

Mode and number of proofs:

```
partition_body_literals(+list,-list,-list,-list,-list) - one
```

---

literal_bucket/2

Classifies a body literal into one of the normalization buckets.

Compilation flags:

```
static
```

Template:

```
literal_bucket(Literal,Bucket)
```

Mode and number of proofs:

```
literal_bucket(+nonvar,-atom) - one
```

Operators

(none)
protocol

1.59.2 datalog_protocol

Datalog and incremental rule engine protocol (stratified negation subset).

Availability:
`logtalk_load(datalog(loader))`

Author: Paulo Moura
Version: 0:1:0
Date: 2026-02-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- Rules: Rules are represented as `rule(Id, Head, Body)` where `Body` is a list of literals using `Term` for positive, `neg(Term)` for negative, and `agg(Op, Template, Goals, Result)` for aggregates where `Op` is one of `count`, `sum`, `min`, or `max`.
- Facts: EDB facts are represented by callable and ground terms.
- Limitations: Current version requires aggregate dependencies to be in lower strata.

Inherited public predicates:
(none)

- *Public predicates*
  - *clear/0*
  - *load_program/1*
  - *add_rule/3*
  - *remove_rule/1*
  - *begin/0*
  - *commit/0*
  - *rollback/0*
  - *assert_fact/1*
  - *retract_fact/1*
  - *materialize/0*
  - *update/3*
  - *query/1*
  - *query/2*
  - *explain/2*

    - *rules/1*
    - *facts/1*
    - *predicate_stratum/3*
    - *strata/1*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

clear/0

Clears all loaded rules, base facts, derived facts, and explanation supports.

Compilation flags:
`static`

Mode and number of proofs:
`clear` - `one`

---

load_program/1

Loads a full program represented as a list of `rule(Id,Head,Body)` and `fact(Fact)` terms, replacing current engine state.

Compilation flags:
`static`

Template:
`load_program(Program)`
Mode and number of proofs:
`load_program(+list)` - `one`

---

add_rule/3

Adds or replaces a rule. Rule safety is checked.

Compilation flags:
`static`

Template:
`add_rule(Id,Head,Body)`
Mode and number of proofs:
`add_rule(+nonvar,+callable,+list(callable))` - `one`

---

remove_rule/1

Removes all rules matching a rule identifier.

Compilation flags:
`static`

Template:
`remove_rule(Id)`
Mode and number of proofs:
`remove_rule(+nonvar)` - `one`

---

begin/0

Starts a transaction by saving the current engine state snapshot.

Compilation flags:
`static`

Mode and number of proofs:
`begin` - `one`

---

commit/0

Commits a transaction by discarding the saved state snapshot.

Compilation flags:
`static`

Mode and number of proofs:
`commit` - `one`

---

rollback/0

Rolls back a transaction by restoring the saved state snapshot.

Compilation flags:
`static`

Mode and number of proofs:
`rollback` - `one`

---

assert_fact/1

Asserts a ground EDB fact if not already present.

Compilation flags:
`static`

Template:
`assert_fact(Fact)`
Mode and number of proofs:
`assert_fact(+callable)` - `one`

---

retract_fact/1

Retracts an EDB fact if present.

Compilation flags:
`static`

Template:
`retract_fact(Fact)`
Mode and number of proofs:
`retract_fact(+callable)` - `one`

---

materialize/0

Computes rule closure from current EDB facts and loaded rules using a fixpoint algorithm.

Compilation flags:
`static`

Mode and number of proofs:
`materialize` - `one`

---

update/3

Applies incremental EDB updates and propagates derivation additions/removals; returns the resulting truth delta.

Compilation flags:
`static`

Template:
`update(Inserts,Deletes,Delta)`

Mode and number of proofs:
`update(+list(callable),+list(callable),-compound)` - `one`

---

query/1

Enumerates currently true facts (EDB + IDB).

Compilation flags:
`static`

Template:
`query(Goal)`
Mode and number of proofs:
`query(?callable)` - `zero_or_more`

---

query/2

Same as `query/1` while returning the unified goal as the second argument.

Compilation flags:
`static`

Template:
`query(Goal,Bindings)`
Mode and number of proofs:
`query(?callable,?callable)` - `zero_or_more`

---

explain/2

Returns one explanation for a currently true fact.

Compilation flags:
`static`

Template:
`explain(Fact,Explanation)`
Mode and number of proofs:
`explain(+callable,-nonvar)` - `zero_or_more`

---

rules/1

Returns the loaded rules.

Compilation flags:
`static`

Template:
`rules(Rules)`
Mode and number of proofs:
`rules(-list)` - `one`

---

facts/1

Returns all currently true facts as a sorted list.

Compilation flags:
`static`

Template:
`facts(Facts)`
Mode and number of proofs:
`facts(-list(callable))` - `one`

---

predicate_stratum/3

Enumerates predicate strata as functor, arity, and stratum number.

Compilation flags:
`static`

Template:
`predicate_stratum(Functor,Arity,Stratum)`
Mode and number of proofs:
`predicate_stratum(?atom,?integer,?integer)` - `zero_or_more`

---

strata/1

Returns all strata grouped by stratum number and contained predicates.

Compilation flags:
`static`

Template:
`strata(Strata)`
Mode and number of proofs:
`strata(-list)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.60 dates

object

1.60.1 date

Date predicates.

Availability:
`logtalk_load(dates(loader))`

Author: Paulo Moura
Version: 2:3:0
Date: 2026-04-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *datep*
Uses:
list
os

Remarks:
(none)

Inherited public predicates:
add_duration/3 after/2 before/2 compare_date_time/3 date_time_to_unix/2 day_of_year/2 day_of_year_date/3 days_in_month/3 duration_between/3 format_date_time/4 leap_year/1 local_to_utc/3 month_weekday_date/5 name_of_day/3 name_of_month/3 normalize_date_time/2 same_instant/2 subtract_duration/3 today/3 unix_to_date_time/2 utc_to_local/3 valid/3 valid_date_time/1 week_of_year_iso/2 weekday/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.60.2 `datep`

Date protocol.

Availability:
`logtalk_load(dates(loader))`

Author: Paulo Moura
Version: 2:3:0
Date: 2026-04-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *today/3*
  - *leap_year/1*
  - *name_of_day/3*
  - *name_of_month/3*
  - *days_in_month/3*
  - *valid/3*
  - *date_time_to_unix/2*
  - *unix_to_date_time/2*
  - *add_duration/3*
  - *subtract_duration/3*
  - *duration_between/3*
  - *utc_to_local/3*
  - *local_to_utc/3*
  - *format_date_time/4*
  - *day_of_year/2*
  - *day_of_year_date/3*
  - *month_weekday_date/5*
  - *week_of_year_iso/2*
  - *weekday/2*
  - *normalize_date_time/2*
  - *valid_date_time/1*
  - *before/2*
  - *after/2*
  - *same_instant/2*
  - *compare_date_time/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

today/3

Returns current date.

Compilation flags:

```
static
```

Template:

```
today(Year,Month,Day)
```

Mode and number of proofs:

```
today(-integer,-integer,-integer) - one
```

---

leap_year/1

True if the argument is a leap year.

Compilation flags:

```
static
```

Template:

```
leap_year(Year)
```

Mode and number of proofs:

```
leap_year(+integer) - zero_or_one
```

---

name_of_day/3

Name and short name of day using ISO weekday numbering (Monday=1, …, Sunday=7).

Compilation flags:

```
static
```

Template:

```
name_of_day(Index,Name,Short)
```

Mode and number of proofs:

```
name_of_day(?integer,?atom,?atom) - zero_or_more
```

---

name_of_month/3

Name and short name of month.

Compilation flags:

```
static
```

Template:

```
name_of_month(Index,Name,Short)
```

Mode and number of proofs:

```
name_of_month(?integer,?atom,?atom) - zero_or_more
```

---

days_in_month/3

Number of days in a month.

Compilation flags:
`static`

Template:
`days_in_month(Month,Year,Days)`
Mode and number of proofs:
`days_in_month(?integer,+integer,?integer)` - `zero_or_more`

---

valid/3

True if the arguments represent a valid date.

Compilation flags:
`static`

Template:
`valid(Year,Month,Day)`
Mode and number of proofs:
`valid(@integer,@integer,@integer)` - `zero_or_one`

---

date_time_to_unix/2

Converts a UTC date-time term `date_time(Year,Month,Day,Hours,Minutes,Seconds)` to Unix epoch seconds.

Compilation flags:
`static`

Template:
`date_time_to_unix(DateTime,UnixTime)`
Mode and number of proofs:
`date_time_to_unix(+compound,-integer)` - `zero_or_one`

---

unix_to_date_time/2

Converts Unix epoch seconds to a UTC date-time term `date_time(Year,Month,Day,Hours,Minutes,Seconds)`.

Compilation flags:
`static`

Template:
`unix_to_date_time(UnixTime,DateTime)`
Mode and number of proofs:
`unix_to_date_time(+integer,-compound)` - `one`

---

add_duration/3

Adds a duration to a datetime. Duration can be integer seconds, `duration(Days,Hours,Minutes,Seconds)`, or a calendar-aware `duration(Years,Months,Days,Hours,Minutes,Seconds)`. For the 6-arity form, the year and month delta is applied first using calendar arithmetic, clamping the day to the last valid day of the resulting month when necessary (e.g. January 31 plus one month gives February 28 or 29), and the remaining day and time delta is then applied via fixed-length arithmetic.

Compilation flags:
`static`

Template:
`add_duration(DateTime,Duration,ResultDateTime)`
Mode and number of proofs:
`add_duration(+compound,+nonvar,-compound)` - `zero_or_one`

---

subtract_duration/3

Subtracts a duration from a datetime. Duration can be integer seconds, `duration(Days,Hours,Minutes,Seconds)`, or a calendar-aware `duration(Years,Months,Days,Hours,Minutes,Seconds)`. For the 6-arity form, the year and month delta is subtracted first using calendar arithmetic with end-of-month day clamping, and the remaining day and time delta is then subtracted via fixed-length arithmetic.

Compilation flags:
`static`

Template:
`subtract_duration(DateTime,Duration,ResultDateTime)`
Mode and number of proofs:
`subtract_duration(+compound,+nonvar,-compound)` - `zero_or_one`

---

duration_between/3

Computes the difference between two datetimes as integer seconds, as `duration(Days,Hours,Minutes,Seconds)`, or as a calendar-aware `duration(Years,Months,Days,Hours,Minutes,Seconds)`. For the 6-arity form, the year and month components count the largest whole number of calendar months between the two datetimes (consistent with the day-clamping semantics of `add_duration/3`), and the remaining days and time fields are the exact residual. For backward intervals all fields are negative.

Compilation flags:

`static`

Template:

`duration_between(StartDateTime,EndDateTime,Duration)`

Mode and number of proofs:

`duration_between(+compound,+compound,?term)` - `zero_or_one`

---

utc_to_local/3

Converts a UTC datetime to a local datetime using an explicit timezone offset atom (`Z` or `+/-HH:MM`).

Compilation flags:

`static`

Template:

`utc_to_local(UTCDateTime,Offset,LocalDateTime)`

Mode and number of proofs:

`utc_to_local(+compound,+atom,-compound)` - `zero_or_one`

---

local_to_utc/3

Converts a local datetime to UTC using an explicit timezone offset atom (`Z` or `+/-HH:MM`).

Compilation flags:

`static`

Template:

`local_to_utc(LocalDateTime,Offset,UTCDateTime)`

Mode and number of proofs:

`local_to_utc(+compound,+atom,-compound)` - `zero_or_one`

---

format_date_time/4

Formats a date-time using an explicit UTC offset in seconds. Supported format identifiers are `rfc3339`, `iso8601`, `atom`, `rfc2822`, `rfc5322`, `rss`, `http_date`, `rfc1123`, `unix_date`, `common_log`, `date_short`, `date_medium`, `date_long`, `date_full`, `time_short`, `time_medium`, `time_long`, `time_full`, `date_time_short`, `date_time_medium`, `date_time_long`, and `date_time_full`. RFC 3339, ISO 8601, Atom, and the `date_*` and `date_time_*` styles require a four-digit non-negative year. Formats that include numeric offsets require an offset expressible in whole minutes. HTTP-date and RFC 1123 output are always normalized to GMT. The style presets are English-only presentation formats. Fails if the format or date-time are not valid.

Compilation flags:
`static`

Template:
`format_date_time(DateTime,OffsetSeconds,Format,String)`
Mode and number of proofs:
`format_date_time(+compound,+integer,+atom,-atom)` - `zero_or_one`

---

day_of_year/2

Computes the day of year (1-366) for a `date(Year,Month,Day)` or `date_time(...)` term.

Compilation flags:
`static`

Template:
`day_of_year(DateLike,DayOfYear)`
Mode and number of proofs:
`day_of_year(+compound,?integer)` - `zero_or_one`

---

day_of_year_date/3

Computes the calendar date corresponding to a year and day of year (1-366) as `date(Year,Month,Day)`.

Compilation flags:
`static`

Template:
`day_of_year_date(Year,DayOfYear,Date)`
Mode and number of proofs:
`day_of_year_date(+integer,+integer,-compound)` - `zero_or_one`

---

month_weekday_date/5

Computes the calendar date for the nth or last weekday in a month as `date(Year,Month,Day)`. Week values 1-4 select the nth weekday and week value 5 selects the last one. Weekday uses ISO numbering (Monday=1, …, Sunday=7).

Compilation flags:

`static`

Template:

`month_weekday_date(Year,Month,Week,Weekday,Date)`

Mode and number of proofs:

`month_weekday_date(+integer,+integer,+integer,+integer,-compound)` - `zero_or_one`

---

week_of_year_iso/2

Computes ISO week for a `date(Year,Month,Day)` or `date_time(...)` term as `week(Week,Year)`.

Compilation flags:

`static`

Template:

`week_of_year_iso(DateLike,ISOWeek)`

Mode and number of proofs:

`week_of_year_iso(+compound,?compound)` - `zero_or_one`

---

weekday/2

Computes ISO weekday number (Monday=1, …, Sunday=7) for a `date(Year,Month,Day)` or `date_time(...)` term.

Compilation flags:

`static`

Template:

`weekday(DateLike,Weekday)`

Mode and number of proofs:

`weekday(+compound,?integer)` - `zero_or_one`

---

normalize_date_time/2

Normalizes a datetime term by carrying overflows/underflows in date and time fields.

Compilation flags:
`static`

Template:
`normalize_date_time(DateTime,NormalizedDateTime)`
Mode and number of proofs:
`normalize_date_time(+compound,-compound)` - `one`

---

valid_date_time/1

True iff a datetime term is valid in strict mode.

Compilation flags:
`static`

Template:
`valid_date_time(DateTime)`
Mode and number of proofs:
`valid_date_time(@compound)` - `zero_or_one`

---

before/2

True iff `DateTime1` represents an instant strictly before `DateTime2`.

Compilation flags:
`static`

Template:
`before(DateTime1,DateTime2)`
Mode and number of proofs:
`before(+compound,+compound)` - `zero_or_one`

---

after/2

True iff DateTime1 represents an instant strictly after DateTime2.

Compilation flags:
static

Template:
after(DateTime1,DateTime2)
Mode and number of proofs:
after(+compound,+compound) - zero_or_one

---

same_instant/2

True iff DateTime1 and DateTime2 represent the same instant (equal Unix epoch seconds).

Compilation flags:
static

Template:
same_instant(DateTime1,DateTime2)
Mode and number of proofs:
same_instant(+compound,+compound) - zero_or_one

---

compare_date_time/3

Three-way comparison of two datetime terms. Order is unified with <, =, or >. Suitable for use with sort/3.

Compilation flags:
static

Template:
compare_date_time(Order,DateTime1,DateTime2)
Mode and number of proofs:
compare_date_time(?atom,+compound,+compound) - zero_or_one

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

date, *timep*

object

1.60.3 `time`

Time predicates.

Availability:
`logtalk_load(dates(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2014-09-27

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *timep*
Uses:
os

Remarks:
(none)

Inherited public predicates:
cpu_time/1 *now/3* *valid/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

datep

protocol

1.60.4 `timep`

Time protocol.

Availability:
`logtalk_load(dates(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2000-07-24

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *now/3*
  - *cpu_time/1*
  - *valid/3*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

now/3

Returns current time.

Compilation flags:
`static`

Template:
`now(Hours,Mins,Secs)`
Mode and number of proofs:
`now(-integer,-integer,-integer) - one`

---

cpu_time/1

Returns the current cpu time.

Compilation flags:
`static`

Template:
`cpu_time(Time)`
Mode and number of proofs:
`cpu_time(-number) - one`

---

valid/3

True if the arguments represent a valid time value.

Compilation flags:
`static`

Template:
`valid(Hours,Mins,Secs)`
Mode and number of proofs:
`valid(+integer,+integer,+integer) - zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

time, *datep*

1.61 dates_tz

object

1.61.1 `dates_tz`

Zone-aware date-time conversion bridging the `dates` and `tzif` libraries.

Availability:
`logtalk_load(dates_tz(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *dates_tz_protocol*
Uses:
date
tzif

Remarks:
(none)

Inherited public predicates:
convert_zones/4 *convert_zones_with_resolution/5* *local_to_utc_tz/3* *local_to_utc_tz_with_resolution/4* *utc_to_local_tz/3*

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dates_tz_protocol, *date*, *tzif*

protocol

1.61.2 `dates_tz_protocol`

Protocol for zone-aware date-time conversion using cached TZif data.

Availability:
```
logtalk_load(dates_tz(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-06-14

Compilation flags:
```
static
```

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *utc_to_local_tz/3*
  - *local_to_utc_tz/3*
  - *local_to_utc_tz_with_resolution/4*
  - *convert_zones/4*
  - *convert_zones_with_resolution/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

utc_to_local_tz/3

Converts a UTC date-time to the civil local date-time in the named zone. Requires the zone to be present in the cached TZif data.

Compilation flags:
`static`

Template:
`utc_to_local_tz(UTCDateTime,Zone,LocalDateTime)`
Mode and number of proofs:
`utc_to_local_tz(+compound,+atom,-compound)` - `one_or_error`

Exceptions:
`Zone` is not present in the cached TZif data:
`existence_error(time_zone,Zone)`

---

local_to_utc_tz/3

Converts a civil local date-time in the named zone to UTC. This strict variant fails silently if the local time falls in a DST gap (non-existent time) or a DST fold (ambiguous time). Requires the zone to be present in the cached TZif data.

Compilation flags:
`static`

Template:
`local_to_utc_tz(LocalDateTime,Zone,UTCDateTime)`
Mode and number of proofs:
`local_to_utc_tz(+compound,+atom,-compound)` - `zero_or_one_or_error`

Exceptions:
`LocalDateTime` is a variable:
`instantiation_error`
`LocalDateTime` is neither a variable nor a valid `date_time/6` term:
`type_error(date_time,LocalDateTime)`

`Zone` is not present in the cached TZif data:
`existence_error(time_zone,Zone)`

`local_to_utc_tz_with_resolution/4`

Converts a civil local date-time in the named zone to UTC using an explicit resolution mode for ambiguous or non-existent times. The resolution mode can be `strict` (fail unless exactly one interpretation), `first` (prefer the earliest valid interpretation), `second` (prefer the latest valid interpretation), or `all` (enumerate all valid interpretations). Requires the zone to be present in the cached TZif data.

Compilation flags:
`static`

Template:
`local_to_utc_tz_with_resolution(LocalDateTime,Zone,ResolutionMode,UTCDateTime)`
Mode and number of proofs:
`local_to_utc_tz_with_resolution(+compound,+atom,+atom,-compound)` - `zero_or_more`

`convert_zones/4`

Converts a civil local date-time in one zone to the civil local date-time in another zone. Uses strict interpretation: fails if the source local time is in a DST gap or fold. Requires both zones to be present in the cached TZif data.

Compilation flags:
`static`

Template:
`convert_zones(LocalDateTime,FromZone,ToZone,ResultDateTime)`
Mode and number of proofs:
`convert_zones(+compound,+atom,+atom,-compound)` - `zero_or_one_or_error`

Exceptions:
`LocalDateTime` is a variable:
`instantiation_error`
`LocalDateTime` is neither a variable nor a valid `date_time/6` term:
`type_error(date_time,LocalDateTime)`
`FromZone` is not present in the cached TZif data:
`existence_error(time_zone,FromZone)`
`ToZone` is not present in the cached TZif data:
`existence_error(time_zone,ToZone)`

---

convert_zones_with_resolution/5

Converts a civil local date-time in one zone to the civil local date-time in another zone using an explicit resolution mode for the source zone. The resolution mode is applied when the source local time is ambiguous or non-existent. Requires both zones to be present in the cached TZif data.

Compilation flags:

```
static
```

Template:

```
convert_zones_with_resolution(LocalDateTime,FromZone,ResolutionMode,ToZone,
ResultDateTime)
```

Mode and number of proofs:

```
convert_zones_with_resolution(+compound,+atom,+atom,+atom,-compound) -
zero_or_more
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

dates_tz

1.62 dbscan_clusterer

object

1.62.1 dbscan_clusterer

DBSCAN clusterer for continuous datasets. Learns from a dataset object implementing the `clustering_dataset_protocol` protocol and returns a clusterer term that can be used for assigning new instances to clusters and exported as predicate clauses.

Availability:

```
logtalk_load(dbscan_clusterer(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *clusterer_common*
`public` *search_indexing*

Uses:

avltree
deque
format
list
numberlist
type

Remarks:

(none)

Inherited public predicates:

check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *kcenters_clusterer*, *kmeans_clusterer*, *kmedoids_clusterer*, *kmedians_clusterer*

1.63 dead_code_scanner

object

1.63.1 `dead_code_scanner`

A tool for detecting *likely* dead code in compiled Logtalk entities and Prolog modules compiled as objects.

Availability:
`logtalk_load(dead_code_scanner(loader))`

Author: Barry Evans and Paulo Moura
Version: 0:18:3
Date: 2026-09-01

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *tool_diagnostics_common*
`public` *options*

Uses:
list
logtalk
os
term_io
type
user

Remarks:

- Dead code: A predicate or non-terminal that is not called (directly or indirectly) by any scoped predicate or non-terminal. These predicates and non-terminals are not used, cannot be called without breaking encapsulation, and are thus considered dead code.
- Diagnostics targets: The diagnostics predicates accept the targets `all`, `entity(Entity)`, `file(File)`, `directory(Directory)`, `rdirectory(Directory)`, `library(Library)`, and `rlibrary(Library)`. The `entity(Entity)` target requires a loaded object or category. The `file(File)` target accepts a loaded source file specified by name, basename, full path, or library notation. Directory and library targets restrict diagnostics to loaded files found in the given directory, recursively in its sub-directories, in the given library, or recursively in its sub-libraries.
- Known issues: Use of local meta-calls with goal arguments only know at runtime can result in false positives. Calls from non-standard meta-predicates may be missed if the meta-calls are not optimized.
- Requirements: Source files must be compiled with the `source_data` flag turned on. To avoid false positives do to meta-calls, compilation of source files with the `optimized` flag turned on is also advised.

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostic/3* *diagnostic_rule/5* *diagnostic_rules/1* *diagnostic_target/1* *diagnostics/2* *diagnostics/3* *diagnostics_preflight/2* *diagnostics_preflight/3* *diagnostics_summary/2* *diagnostics_summary/3* *diagnostics_tool/5* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *entity/1*
  - *entity/2*
  - *file/2*
  - *file/1*
  - *directory/2*
  - *directory/1*
  - *rdirectory/2*
  - *rdirectory/1*
  - *library/2*
  - *library/1*
  - *rlibrary/2*
  - *rlibrary/1*
  - *all/1*
  - *all/0*
  - *predicates/2*
  - *predicates/3*
  - *predicate/2*
  - *predicate/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

entity/1

Scans a loaded entity for dead code. Fails if the entity does not exist.

Compilation flags:

```
static
```

Template:

```
entity(Entity)
```

Mode and number of proofs:

```
entity(+entity_identifier) - zero_or_one
```

entity/2

Scans a loaded entity for dead code using the given options. Fails if the entity does not exist.

Compilation flags:

```
static
```

Template:

```
entity(Entity,Options)
```

Mode and number of proofs:

```
entity(+entity_identifier,+list(compound)) - zero_or_one
```

---

file/2

Scans all entities in a loaded source file for dead code using the given options. The file can be given by name, basename, full path, or using library notation. Fails if the file is not loaded.

Compilation flags:

```
static
```

Template:

```
file(File,Options)
```

Mode and number of proofs:

```
file(+atom,+list(compound)) - zero_or_one
```

---

file/1

Scans all entities in a loaded source file for dead code using default options. The file can be given by name, basename, full path, or using library notation. Fails if the file is not loaded.

Compilation flags:

```
static
```

Template:

```
file(File)
```

Mode and number of proofs:

```
file(+atom) - zero_or_one
```

---

directory/2

Scans all entities in all loaded files from a given directory for dead code using the given options.

Compilation flags:
```
static
```

Template:
```
directory(Directory,Options)
```
Mode and number of proofs:
```
directory(+atom,+list(compound)) - one
```

---

directory/1

Scans all entities in all loaded files from a given directory for dead code using default options.

Compilation flags:
```
static
```

Template:
```
directory(Directory)
```
Mode and number of proofs:
```
directory(+atom) - one
```

---

rdirectory/2

Scans all entities in all loaded files from a given directory and its sub-directories for dead code using the given options.

Compilation flags:
```
static
```

Template:
```
rdirectory(Directory,Options)
```
Mode and number of proofs:
```
rdirectory(+atom,+list(compound)) - one
```

---

rdirectory/1

Scans all entities in all loaded files from a given directory and its sub-directories for dead code using default options.

Compilation flags:
```
static
```

Template:
```
rdirectory(Directory)
```
Mode and number of proofs:
```
rdirectory(+atom) - one
```

---

library/2

Scans all entities in all loaded files from a given library for dead code using the given options.

Compilation flags:
```
static
```

Template:
```
library(Library,Options)
```
Mode and number of proofs:
```
library(+atom,+list(compound)) - one
```

---

library/1

Scans all entities in all loaded files from a given library for dead code using default options.

Compilation flags:
```
static
```

Template:
```
library(Library)
```
Mode and number of proofs:
```
library(+atom) - one
```

---

rlibrary/2

Scans all entities in all loaded files in a loaded library and its sub-libraries for dead code using the given options.

Compilation flags:
`static`

Template:
`rlibrary(Library,Options)`
Mode and number of proofs:
`rlibrary(+atom,+list(compound))` - `one`

---

rlibrary/1

Scans all entities in all loaded files in a loaded library and its sub-libraries for dead code using default options.

Compilation flags:
`static`

Template:
`rlibrary(Library)`
Mode and number of proofs:
`rlibrary(+atom)` - `one`

---

all/1

Scans all entities for dead code using the given options.

Compilation flags:
`static`

Template:
`all(Options)`
Mode and number of proofs:
`all(+list(compound))` - `one`

---

all/0

Scans all entities for dead code using default options.

Compilation flags:

```
static
```

Mode and number of proofs:

```
all - one
```

---

predicates/2

Returns an ordered set of local predicates (and non-terminals) that are not used, directly or indirectly, by scoped predicates for a loaded entity.

Compilation flags:

```
static
```

Template:

```
predicates(Entity,Predicates)
```

Mode and number of proofs:

```
predicates(+entity_identifier,-list(predicate_indicator)) - one
```

---

predicates/3

Returns an ordered set of local predicates (and non-terminals) that are not used, directly or indirectly, by scoped predicates for a loaded entity using the given options.

Compilation flags:

```
static
```

Template:

```
predicates(Entity,Predicates,Options)
```

Mode and number of proofs:

```
predicates(+entity_identifier,-list(predicate_indicator),+list(compound)) - one
```

---

predicate/2

Enumerates, by backtracking, local predicates (and non-terminals) that are not used, directly or indirectly, by scoped predicates for a loaded entity.

Compilation flags:
`static`

Template:
`predicate(Entity,Predicate)`
Mode and number of proofs:
`predicate(+entity_identifier,?predicate_indicator)` - `zero_or_more`

---

predicate/3

Enumerates, by backtracking, local predicates (and non-terminals) that are not used, directly or indirectly, by scoped predicates for a loaded entity using the given options.

Compilation flags:
`static`

Template:
`predicate(Entity,Predicate,Options)`
Mode and number of proofs:
`predicate(+entity_identifier,?predicate_indicator,+list(compound))` - `zero_or_more`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.63.2 dead_code_scanner_messages

Logtalk `dead_code_scanner` tool default message translations.

Availability:
`logtalk_load(dead_code_scanner(loader))`

Author: Barry Evans and Paulo Moura

Version: 0:9:1
Date: 2026-05-05

Compilation flags:
`static`

Provides:
logtalk::message_prefix_stream/4
logtalk::message_tokens//2

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.64 debug_messages

object

1.64.1 debug_messages

Supports selective enabling and disabling of `debug` and `debug(Group)` messages.

Availability:
`logtalk_load(debug_messages(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2022-05-05

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
logtalk

Remarks:

- Limitations: Debug messages are suppressed by the compiler when the `optimize` flag is turned on and thus cannot be enabled in this case.

Inherited public predicates:
(none)

- *Public predicates*
  - *enable/1*
  - *disable/1*
  - *enabled/1*
  - *enable/2*
  - *disable/2*
  - *enabled/2*
- *Protected predicates*
- *Private predicates*
  - *enabled_/1*
  - *enabled_/2*
- *Operators*

Public predicates

enable/1

Enables all `debug` and `debug(Group)` messages for the given component.

Compilation flags:
`static`

Template:
`enable(Component)`
Mode and number of proofs:
`enable(@term)` - `one`

---

disable/1

Disables all `debug` and `debug(Group)` messages for the given component.

Compilation flags:
`static`

Template:
`disable(Component)`
Mode and number of proofs:
`disable(@term)` - `one`

---

enabled/1

Enumerates by backtracking the components with enabled `debug` and `debug(Group)` messages.

Compilation flags:
`static`

Template:
`enabled(Component)`
Mode and number of proofs:
`enabled(?term)` - `zero_or_more`

---

enable/2

Enables `debug(Group)` messages for the given component and group.

Compilation flags:
`static`

Template:
`enable(Component,Group)`
Mode and number of proofs:
`enable(@term,@term)` - `one`

---

disable/2

Disables `debug(Group)` messages for the given component and group.

Compilation flags:
`static`

Template:
`disable(Component,Group)`
Mode and number of proofs:
`disable(@term,@term)` - `one`

---

enabled/2

Enumerates by backtracking the enabled `debug(Group)` messages for each component.

Compilation flags:
`static`

Template:
`enabled(Component,Group)`
Mode and number of proofs:
`enabled(?term,?term)` - `zero_or_more`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

enabled_/1

Table of components with currently enabled `debug` and `debug(Group)` messages.

Compilation flags:
`dynamic`

Template:
`enabled_(Component)`
Mode and number of proofs:
`enabled_(?term)` - `zero_or_more`

---

enabled_/2

Table of currently enabled `debug(Group)` per component.

Compilation flags:
`dynamic`

Template:
`enabled_(Component,Group)`
Mode and number of proofs:
`enabled_(?term,?term)` - `zero_or_more`

Operators

(none)

1.65 debugger

object

1.65.1 debugger

Command-line debugger based on an extended procedure box model supporting execution tracing and spy points.

Availability:
`logtalk_load(debugger(loader))`

Author: Paulo Moura

Version: 8:1:0
Date: 2026-11-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *debuggerp*
Provides:
logtalk::debug_handler/1
logtalk::debug_handler/3
Uses:
logtalk

Remarks:
(none)

Inherited public predicates:
debug/0 debugging/0 debugging/1 leash/1 leashing/1 log/3 logging/3 nodebug/0 nolog/3 nologall/0 nospy/1 nospy/3 nospy/4 nospyall/0 notrace/0 reset/0 set_write_max_depth/1 spy/1 spy/3 spy/4 spying/1 spying/3 spying/4 trace/0 write_max_depth/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *debugging_/0*
  - *tracing_/0*
  - *explicit_tracing_/0*
  - *skipping_/0*
  - *skipping_unleashed_/1*
  - *quasi_skipping_/0*
  - *leaping_/1*
  - *leashing_/1*
  - *invocation_number_/1*
  - *jump_to_invocation_number_/1*
  - *zap_to_port_/1*
  - *write_max_depth_/1*
  - *log_point_/3*
  - *clause_breakpoint_/2*
  - *predicate_breakpoint_/3*
  - *entity_predicate_breakpoint_/4*
  - *context_breakpoint_/4*
  - *conditional_breakpoint_/3*
  - *triggered_breakpoint_/4*
  - *triggered_breakpoint_enabled_/2*
  - *file_line_hit_count_/3*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

debugging_/0

True iff debug is on.

Compilation flags:
`dynamic`

Mode and number of proofs:
`debugging_` - `zero_or_one`

---

tracing_/0

True iff tracing is on.

Compilation flags:
`dynamic`

Mode and number of proofs:
`tracing_` - `zero_or_one`

---

explicit_tracing_/0

True iff tracing is on due to a call to the trace/0 predicate.

Compilation flags:
`dynamic`

Mode and number of proofs:
`explicit_tracing_` - `zero_or_one`

---

skipping_/0

True iff skipping.

Compilation flags:
dynamic

Mode and number of proofs:
skipping_ - zero_or_one

---

skipping_unleashed_/1

True iff skipping (a goal with invocation number N) but showing intermediate ports as unleashed.

Compilation flags:
dynamic

Template:
skipping_unleashed_(N)
Mode and number of proofs:
skipping_unleashed_(?integer) - zero_or_one

---

quasi_skipping_/0

True iff quasi-skipping.

Compilation flags:
dynamic

Mode and number of proofs:
quasi_skipping_ - zero_or_one

---

leaping_/1

True iff leaping in tracing or debugging mode.

Compilation flags:
dynamic

Template:
leaping_(Mode)
Mode and number of proofs:
leaping_(?atom) - zero_or_one

---

leashing_/1

Table of currently leashed ports.

Compilation flags:
`dynamic`

Template:
`leashing_(Port)`
Mode and number of proofs:
`leashing_(?atom)` - `zero_or_more`

---

invocation_number_/1

Current call stack invocation number.

Compilation flags:
`dynamic`

Template:
`invocation_number_(N)`
Mode and number of proofs:
`invocation_number_(?integer)` - `zero_or_one`

---

jump_to_invocation_number_/1

Invocation number to jump to.

Compilation flags:
`dynamic`

Template:
`jump_to_invocation_number_(N)`
Mode and number of proofs:
`jump_to_invocation_number_(?integer)` - `zero_or_one`

---

zap_to_port_/1

Port to zap to.

Compilation flags:
`dynamic`

Template:
`zap_to_port_(Port)`
Mode and number of proofs:
`zap_to_port_(?integer)` - `zero_or_one`

---

write_max_depth_/1

Current term write maximum depth.

Compilation flags:
`dynamic`

Template:
`write_max_depth_(MaxDepth)`
Mode and number of proofs:
`write_max_depth_(?non_negative_integer)` - `zero_or_one`

---

log_point_/3

Table of log points.

Compilation flags:
`dynamic`

Template:
`log_point_(Entity,Line,Message)`
Mode and number of proofs:
`log_point_(?object_identifier,?integer,?atom)` - `zero_or_more`
`log_point_(?category_identifier,?integer,?atom)` - `zero_or_more`

---

clause_breakpoint_/2

Table of clause breakpoints.

Compilation flags:
```
dynamic
```

Template:
```
clause_breakpoint_(Entity,Line)
```
Mode and number of proofs:
```
clause_breakpoint_(?object_identifier,?integer) - zero_or_more
clause_breakpoint_(?category_identifier,?integer) - zero_or_more
```

---

predicate_breakpoint_/3

Table of predicate breakpoints.

Compilation flags:
```
dynamic
```

Template:
```
predicate_breakpoint_(Functor,Arity,Original)
```
Mode and number of proofs:
```
predicate_breakpoint_(?atom,?integer,?predicate_indicator) - zero_or_more
predicate_breakpoint_(?atom,?integer,?non_terminal_indicator) - zero_or_more
```

---

entity_predicate_breakpoint_/4

Table of entity predicate breakpoints.

Compilation flags:
```
dynamic
```

Template:
```
entity_predicate_breakpoint_(Entity,Functor,Arity,Original)
```
Mode and number of proofs:
```
entity_predicate_breakpoint_(?callable,?atom,?integer,?
qualified_predicate_indicator) - zero_or_more
entity_predicate_breakpoint_(?callable,?atom,?integer,?
qualified_non_terminal_indicator) - zero_or_more
```

---

context_breakpoint_/4

Table of context breakpoints.

Compilation flags:
dynamic

Template:
context_breakpoint_(Sender,This,Self,Goal)

Mode and number of proofs:
context_breakpoint_(?object_identifier,?object_identifier,?object_identifier,?callable) - zero_or_more

---

conditional_breakpoint_/3

Table of conditional breakpoints.

Compilation flags:
dynamic

Template:
conditional_breakpoint_(Entity,Line,Condition)

Mode and number of proofs:
conditional_breakpoint_(?object_identifier,?integer,?callable) - zero_or_more
conditional_breakpoint_(?category_identifier,?integer,?callable) - zero_or_more

---

triggered_breakpoint_/4

Table of defined triggered breakpoints.

Compilation flags:
dynamic

Template:
triggered_breakpoint_(Entity,Line,TriggerEntity,TriggerLine)

Mode and number of proofs:
triggered_breakpoint_(?object_identifier,?integer,?object_identifier,?integer) - zero_or_more
triggered_breakpoint_(?object_identifier,?integer,?category_identifier,?integer) - zero_or_more
triggered_breakpoint_(?category_identifier,?integer,?object_identifier,?integer) - zero_or_more
triggered_breakpoint_(?category_identifier,?integer,?category_identifier,?integer) - zero_or_more

---

triggered_breakpoint_enabled_/2

Table of enabled triggered breakpoints.

Compilation flags:
`dynamic`

Template:
`triggered_breakpoint_enabled_(Entity,Line)`
Mode and number of proofs:
`triggered_breakpoint_enabled_(?object_identifier,?integer)` - `zero_or_more`
`triggered_breakpoint_enabled_(?category_identifier,?integer)` - `zero_or_more`

---

file_line_hit_count_/3

Table of file and line hit counts (successful unifications with clause heads).

Compilation flags:
`dynamic`

Template:
`file_line_hit_count_(File,Line,Count)`
Mode and number of proofs:
`file_line_hit_count_(?atom,?integer,?integer)` - `zero_or_one`

Operators

(none)
category

1.65.2 debugger_messages

Logtalk `debugger` tool default message translations.

Availability:
`logtalk_load(debugger(loader))`

Author: Paulo Moura
Version: 4:0:0
Date: 2025-12-18

Compilation flags:
`static`

Provides:

logtalk::message_prefix_stream/4
logtalk::question_prompt_stream/4
logtalk::message_tokens//2

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.65.3 `debuggerp`

Debugger protocol.

Availability:
`logtalk_load(debugger(loader))`

Author: Paulo Moura
Version: 3:6:0
Date: 2025-09-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- Debugger help: Type the character `h` (condensed help) or the character `?` (extended help) at a leashed port.
- Predicate breakpoint: Specified as a ground term `Functor/Arity`.
- Non-terminal breakpoint: Specified as a ground term `Functor//Arity`.
- Entity predicate breakpoint: Specified as a term `Entity::Functor/Arity`. `Entity` must be an object or category and may not be ground if parametric.
- Entity non-terminal breakpoint: Specified as a term `Entity::Functor//Arity`. `Entity` must be an object or category and may not be ground if parametric.
- Clause breakpoint: Specified as an `Entity-Line` term with both `Entity` and `Line` bound. `Line` must be the first source file line of an entity clause.
- Conditional breakpoint: Specified using `Entity` and `Line` for the clause head and a condition. `Line` must be the first source file line of an entity clause.
- Hit count breakpoint: Specified using `Entity` and `Line` for the clause head and an unification count expression as a condition. `Line` must be the first source file line of an entity clause.
- Triggered breakpoint: Specified using `Entity` and `Line` for the clause head and another breakpoint as a condition. `Line` must be the first source file line of an entity clause.
- Context breakpoint: Specified as a `(Sender, This, Self, Goal)` tuple.
- Log point: Specified using `Entity` and `Line` for the clause head and a message.
- Leash port shorthands: `none` - `[]`, `loose` - `[fact,rule,call]`, `half` - `[fact,rule,call,redo]`, `tight` - `[fact,rule,call,redo,fail,exception]`, and `full` - `[fact,rule,call,exit,redo,fail,exception]`.

Inherited public predicates:

(none)

- *Public predicates*
  - *reset/0*
  - *debug/0*
  - *nodebug/0*
  - *debugging/0*
  - *debugging/1*
  - *trace/0*
  - *notrace/0*
  - *leash/1*
  - *leashing/1*
  - *spy/1*
  - *spying/1*
  - *nospy/1*
  - *spy/3*
  - *spying/3*
  - *nospy/3*
  - *spy/4*
  - *spying/4*
  - *nospy/4*
  - *nospyall/0*
  - *log/3*
  - *logging/3*

    - *nolog/3*
    - *nologall/0*
    - *write_max_depth/1*
    - *set_write_max_depth/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

reset/0

Resets all debugging settings (including breakpoints, log points, and leashed ports) and turns off debugging.

Compilation flags:
`static`

Mode and number of proofs:
`reset` - `one`

See also:
nospyall/0

---

debug/0

Starts debugging for all defined breakpoints.

Compilation flags:
`static`

Mode and number of proofs:
`debug` - `one`

---

nodebug/0

Stops debugging for all defined breakpoints. Also turns off tracing. Does not remove defined breakpoints.

Compilation flags:
`static`

Mode and number of proofs:
`nodebug` - `one`

See also:

reset/0

---

debugging/0

Reports current debugging settings, including breakpoints and log points.

Compilation flags:
`static`

Mode and number of proofs:
`debugging` - `one`

---

debugging/1

Enumerates, by backtracking, all entities compiled in debug mode.

Compilation flags:
`static`

Template:
`debugging(Entity)`
Mode and number of proofs:
`debugging(?entity_identifier)` - `zero_or_more`

---

trace/0

Starts tracing all calls compiled in debug mode.

Compilation flags:
`static`

Mode and number of proofs:
`trace` - `one`

---

notrace/0

Stops tracing of calls compiled in debug mode. Debugger will still stop at defined breakpoints.

Compilation flags:
`static`

Mode and number of proofs:
`notrace` - `one`

---

leash/1

Sets the debugger leash ports using an abbreviation (`none`, `loose`, `half`, `tight`, or `full`) or a list of ports (valid ports are `fact`, `rule`, `call`, `exit`, `redo`, `fail`, and `exception`).

Compilation flags:
`static`

Template:
`leash(Ports)`
Mode and number of proofs:
`leash(+atom)` - `one`
`leash(+list(atom))` - `one`

---

leashing/1

Enumerates, by backtracking, all leashed ports (valid ports are `fact`, `rule`, `call`, `exit`, `redo`, `fail`, and `exception`).

Compilation flags:
`static`

Template:
`leashing(Port)`
Mode and number of proofs:
`leashing(?atom)` - `zero_or_more`

---

spy/1

Sets a predicate or clause breakpoint (removing any existing log point or breakpoint defined for the same location, or a list of breakpoints. Fails if a breakpoint is invalid.

Compilation flags:

```
static
```

Template:

```
spy(Breakpoint)
```

Mode and number of proofs:

```
spy(@spy_point) - zero_or_one
spy(@list(spy_point)) - zero_or_one
```

---

spying/1

Enumerates, by backtracking, all defined predicate and clause breakpoints.

Compilation flags:

```
static
```

Template:

```
spying(Breakpoint)
```

Mode and number of proofs:

```
spying(?spy_point) - zero_or_more
```

---

nospy/1

Removes all matching predicate and clause breakpoints.

Compilation flags:

```
static
```

Template:

```
nospy(Breakpoint)
```

Mode and number of proofs:

```
nospy(@var) - one
nospy(@spy_point) - one
nospy(@list(spy_point)) - one
```

---

spy/3

Sets a conditional or triggered breakpoint (removing any existing log point or breakpoint defined for the same location) at a clause head. The condition can be a unification count expression, a lambda expression, or another breakpoint. Fails if the breakpoint is invalid.

Compilation flags:
`static`

Template:
`spy(Entity,Line,Condition)`

Mode and number of proofs:
`spy(+atom,+integer,@callable)` - `zero_or_one`

Remarks:

- Unification count expression conditions: `>(Count)`, `>=(Count)`, `=:=(Count)`, `=<(Count)`, `<(Count)`, `mod(M)`, and `Count`.
- Lambda expression conditions: `[Count,N,Goal]>>Condition` and `[Goal]>>Condition` where `Count` is the unification count, `N` is the invocation number, and `Goal` is the goal that unified with the clause head; `Condition` is called in the context of `user`.
- Triggered breakpoint conditions: `Entity-Line`.

---

spying/3

Enumerates, by backtracking, all conditional and triggered breakpoints.

Compilation flags:
`static`

Template:
`spying(Entity,Line,Condition)`

Mode and number of proofs:
`spying(?atom,?integer,?callable)` - `zero_or_more`

---

nospy/3

Removes all matching conditional and triggered breakpoints.

Compilation flags:
`static`

Template:
`nospy(Entity,Line,Condition)`

Mode and number of proofs:
`nospy(@term,@term,@term)` - `one`

---

spy/4

Sets a context breakpoint.

Compilation flags:
`static`

Template:
`spy(Sender,This,Self,Goal)`
Mode and number of proofs:
`spy(@term,@term,@term,@term) - one`

---

spying/4

Enumerates, by backtracking, all defined context breakpoints.

Compilation flags:
`static`

Template:
`spying(Sender,This,Self,Goal)`
Mode and number of proofs:
`spying(?term,?term,?term,?term) - zero_or_more`

---

nospy/4

Removes all matching context breakpoints.

Compilation flags:
`static`

Template:
`nospy(Sender,This,Self,Goal)`
Mode and number of proofs:
`nospy(@term,@term,@term,@term) - one`

---

nospyall/0

Removes all breakpoints and log points.

Compilation flags:
`static`

Mode and number of proofs:
`nospyall` - `one`

See also:
reset/0

---

log/3

Sets a log point (removing any existing breakpoint defined for the same location) at a clause head. Fails if the log point is invalid.

Compilation flags:
`static`

Template:
`log(Entity,Line,Message)`
Mode and number of proofs:
`log(@object_identifier,+integer,+atom)` - `zero_or_one`
`log(@category_identifier,+integer,+atom)` - `zero_or_one`

---

logging/3

Enumerates, by backtracking, all defined log points.

Compilation flags:
`static`

Template:
`logging(Entity,Line,Message)`
Mode and number of proofs:
`logging(?object_identifier,?integer,?atom)` - `zero_or_more`
`logging(?category_identifier,?integer,?atom)` - `zero_or_more`

---

nolog/3

Removes all matching log points.

Compilation flags:
`static`

Template:
`nolog(Entity,Line,Message)`
Mode and number of proofs:
`nolog(@var_or(object_identifier),@var_or(integer),@var_or(atom))` - `one`
`nolog(@var_or(category_identifier),@var_or(integer),@var_or(atom))` - `one`

---

nologall/0

Removes all log points.

Compilation flags:
`static`

Mode and number of proofs:
`nologall` - `one`

See also:
reset/0

---

write_max_depth/1

Current term write maximum depth. When not defined, the backend default is used.

Compilation flags:
`static`

Template:
`write_max_depth(MaxDepth)`
Mode and number of proofs:
`write_max_depth(?non_negative_integer)` - `zero_or_one`

`set_write_max_depth/1`

Sets the default term maximum write depth. For most backends, a value of zero means that the whole term is written.

Compilation flags:
`static`

Template:
`set_write_max_depth(MaxDepth)`
Mode and number of proofs:
`set_write_max_depth(+non_negative_integer)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

debugger

object

1.65.4 `dump_trace`

Simple solution for redirecting a debugger trace to a file.

Availability:
`logtalk_load(debugger(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2021-11-12

Compilation flags:
`static, context_switching_calls`

Uses:
debugger

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *start_redirect_to_file/2*
  - *stop_redirect_to_file/0*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

start_redirect_to_file/2

Starts redirecting debugger trace messages to a file.

Compilation flags:
`static`

Template:
`start_redirect_to_file(File,Goal)`
Meta-predicate template:
`start_redirect_to_file(*,0)`
Mode and number of proofs:
`start_redirect_to_file(+atom,+callable)` - `zero_or_more`

---

stop_redirect_to_file/0

Stops redirecting debugger trace messages to a file.

Compilation flags:
`static`

Mode and number of proofs:
`stop_redirect_to_file` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.66 dependents

category

1.66.1 `observer`

Smalltalk dependent protocol.

Availability:
`logtalk_load(dependents(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2003-02-09

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - `update/1`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

update/1

Called when an observed object is updated.

Compilation flags:
`static`

Template:
`update(Change)`
Mode and number of proofs:
`update(?nonvar)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

subject

category

1.66.2 subject

Smalltalk dependent handling predicates.

Availability:
`logtalk_load(dependents(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2003-02-09

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *changed/0*
  - *changed/1*
  - *dependents/1*
  - *addDependent/1*
  - *removeDependent/1*
- *Protected predicates*
- *Private predicates*
  - *dependent_/1*
- *Operators*

Public predicates

changed/0

Receiver changed in some way. Notify all dependents.

Compilation flags:
`static`

Mode and number of proofs:
`changed` - `one`

---

changed/1

Receiver changed as specified in the argument. Notify all dependents.

Compilation flags:
`static`

Template:
`changed(Change)`
Mode and number of proofs:
`changed(?nonvar)` - `one`

---

dependents/1

Returns a list of all dependent objects.

Compilation flags:
`static`

Template:
`dependents(Dependents)`
Mode and number of proofs:
`dependents(-list)` - `one`

---

addDependent/1

Adds a new dependent object.

Compilation flags:
`static`

Template:
`addDependent(Dependent)`
Mode and number of proofs:
`addDependent(@object)` - `one`

---

removeDependent/1

Removes a dependent object.

Compilation flags:
`static`

Template:
`removeDependent(Dependent)`
Mode and number of proofs:
`removeDependent(?object)` - `zero_or_more`

Protected predicates

(none)

Private predicates

dependent_/1

Table of dependent objects.

Compilation flags:
`dynamic`

Template:
`dependent_(Dependent)`
Mode and number of proofs:
`dependent_(?object)` - `zero_or_more`

Operators

(none)

See also

observer

1.67 deques

object

1.67.1 deque

Double-ended queue (deque) implementation using difference lists to provide O(1) operations at both ends.

Availability:
`logtalk_load(deques(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-09

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *deque_protocol*
Extends:
`public` *compound*

Uses:
list

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_deque/2 as_list/2 check/1 depth/2 empty/1 ground/1 length/2 map/2 map/3 new/1 numbervars/1 numbervars/3 occurs/2 peek_back/2 peek_front/2 pop_back/3 pop_front/3 push_back/3 push_front/3 singletons/2 subsumes/2 subterm/2 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.67.2 `deque_protocol`

Extracted protocol entity.

Availability:
`logtalk_load(deques(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *empty/1*
  - *push_front/3*
  - *push_back/3*
  - *pop_front/3*
  - *pop_back/3*
  - *peek_front/2*
  - *peek_back/2*
  - *length/2*
  - *map/2*
  - *map/3*
  - *as_list/2*
  - *as_deque/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

empty/1

True iff the deque is empty.

Compilation flags:
`static`

Template:
`empty(Deque)`

Mode and number of proofs:
`empty(+deque)` - `zero_or_one`

push_front/3

Adds an element to the front of the deque.

Compilation flags:
`static`

Template:
`push_front(Element,DequeIn,DequeOut)`
Mode and number of proofs:
`push_front(+term,+deque,-deque) - one`

---

push_back/3

Adds an element to the back of the deque.

Compilation flags:
`static`

Template:
`push_back(Element,DequeIn,DequeOut)`
Mode and number of proofs:
`push_back(+term,+deque,-deque) - one`

---

pop_front/3

Removes and returns the front element.

Compilation flags:
`static`

Template:
`pop_front(DequeIn,Element,DequeOut)`
Mode and number of proofs:
`pop_front(+deque,-term,-deque) - zero_or_one`

---

pop_back/3

Removes and returns the back element.

Compilation flags:
`static`

Template:
`pop_back(DequeIn,Element,DequeOut)`
Mode and number of proofs:
`pop_back(+deque,-term,-deque)` - `zero_or_one`

---

peek_front/2

Returns the front element without removing it.

Compilation flags:
`static`

Template:
`peek_front(Deque,Element)`
Mode and number of proofs:
`peek_front(+deque,-term)` - `zero_or_one`

---

peek_back/2

Returns the back element without removing it.

Compilation flags:
`static`

Template:
`peek_back(Deque,Element)`
Mode and number of proofs:
`peek_back(+deque,-term)` - `zero_or_one`

---

length/2

Returns the number of elements in the deque.

Compilation flags:
```
static
```

Template:
```
length(Deque,Length)
```
Mode and number of proofs:
```
length(+deque,-integer) - one
```

---

map/2

Applies a closure to all elements of a deque.

Compilation flags:
```
static
```

Template:
```
map(Closure,Deque)
```
Meta-predicate template:
```
map(1,*)
```
Mode and number of proofs:
```
map(+callable,+deque) - zero_or_one
```

---

map/3

Applies a closure to all elements of a deque constructing a new deque.

Compilation flags:
```
static
```

Template:
```
map(Closure,Deque,NewQueue)
```
Meta-predicate template:
```
map(2,*,*)
```
Mode and number of proofs:
```
map(+callable,+deque,?deque) - zero_or_one
```

---

as_list/2

Converts a deque to a list.

Compilation flags:
`static`

Template:
`as_list(Deque,List)`
Mode and number of proofs:
`as_list(+deque,-list)` - `one`

---

as_deque/2

Converts a list to a deque.

Compilation flags:
`static`

Template:
`as_deque(List,Deque)`
Mode and number of proofs:
`as_deque(+list,-deque)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.68 derangements

object

1.68.1 derangements

Implementation of derangement operations over lists.

Availability:
`logtalk_load(derangements(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *derangements_protocol*
Uses:
fast_random(Algorithm)
list
natural
permutations

Remarks:
(none)

Inherited public predicates:
count_derangements/2 *count_distinct_derangements/2* *count_partial_derangements/3* *derangement/2* *derangement/3* *derangement_index/3* *derangement_index/4* *derangements/2* *derangements/3* *distinct_derangement/2* *distinct_derangement/3* *distinct_derangement_index/3* *distinct_derangements/2* *distinct_derangements/3* *next_derangement/3* *nth_derangement/3* *nth_derangement/4* *nth_distinct_derangement/3* *previous_derangement/3* *random_derangement/2* *random_distinct_derangement/2* *sample_derangements/3* *sample_distinct_derangements/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.68.2 derangements_protocol

Protocol for derangement operations over lists.

Availability:
`logtalk_load(derangements(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-12

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *derangements/2*
  - *derangement/2*
  - *derangements/3*
  - *derangement/3*
  - *distinct_derangements/2*
  - *distinct_derangement/2*
  - *distinct_derangements/3*
  - *distinct_derangement/3*

      - *nth_derangement/3*
      - *nth_derangement/4*
      - *derangement_index/3*
      - *derangement_index/4*
      - *count_derangements/2*
      - *count_partial_derangements/3*
      - *count_distinct_derangements/2*
      - *nth_distinct_derangement/3*
      - *distinct_derangement_index/3*
      - *random_derangement/2*
      - *sample_derangements/3*
      - *random_distinct_derangement/2*
      - *sample_distinct_derangements/3*
      - *next_derangement/3*
      - *previous_derangement/3*
    - *Protected predicates*
    - *Private predicates*
    - *Operators*

Public predicates

derangements/2

Generates all derangements of a list using default order.

Compilation flags:
`static`

Template:
`derangements(List,Derangements)`
Mode and number of proofs:
`derangements(+list,-list)` - `one`

---

derangement/2

True iff the second argument is a derangement of the first argument using default order.

Compilation flags:
`static`

Template:
`derangement(List,Derangement)`
Mode and number of proofs:
`derangement(+list,-list)` - `zero_or_more`

---

derangements/3

Generates all derangements with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`derangements(List,Order,Derangements)`
Mode and number of proofs:
`derangements(+list,+atom,-list)` - `one`

---

derangement/3

True iff the third argument is a derangement with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`derangement(List,Order,Derangement)`
Mode and number of proofs:
`derangement(+list,+atom,-list)` - `zero_or_more`

---

distinct_derangements/2

Generates all distinct derangements of a list using default order.

Compilation flags:
`static`

Template:
`distinct_derangements(List,Derangements)`
Mode and number of proofs:
`distinct_derangements(+list,-list)` - `one`

---

distinct_derangement/2

True iff the second argument is a distinct derangement of the first argument using default order.

Compilation flags:
`static`

Template:
`distinct_derangement(List,Derangement)`
Mode and number of proofs:
`distinct_derangement(+list,-list)` - `zero_or_more`

---

distinct_derangements/3

Generates all distinct derangements with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`distinct_derangements(List,Order,Derangements)`
Mode and number of proofs:
`distinct_derangements(+list,+atom,-list)` - `one`

---

distinct_derangement/3

True iff the third argument is a distinct derangement with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`distinct_derangement(List,Order,Derangement)`
Mode and number of proofs:
`distinct_derangement(+list,+atom,-list)` - `zero_or_more`

---

nth_derangement/3

Returns the derangement at a given zero-based index using default order.

Compilation flags:
`static`

Template:
`nth_derangement(List,Index,Derangement)`
Mode and number of proofs:
`nth_derangement(+list,+integer,-list)` - `zero_or_one`

---

nth_derangement/4

Returns the derangement at a given zero-based index in the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`nth_derangement(List,Order,Index,Derangement)`
Mode and number of proofs:
`nth_derangement(+list,+atom,+integer,-list)` - `zero_or_one`

---

derangement_index/3

Returns the zero-based index of a derangement using default order.

Compilation flags:
`static`

Template:
`derangement_index(List,Derangement,Index)`
Mode and number of proofs:
`derangement_index(+list,+list,-integer)` - `zero_or_one`

---

derangement_index/4

Returns the zero-based index of a derangement in the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`derangement_index(List,Order,Derangement,Index)`
Mode and number of proofs:
`derangement_index(+list,+atom,+list,-integer)` - `zero_or_one`

---

count_derangements/2

Counts the number of derangements of a list.

Compilation flags:
`static`

Template:
`count_derangements(List,Count)`
Mode and number of proofs:
`count_derangements(+list,-integer)` - `one`

---

count_partial_derangements/3

Counts the number of permutations of a list with exactly the given number of fixed points.

Compilation flags:
`static`

Template:
`count_partial_derangements(FixedPoints,List,Count)`
Mode and number of proofs:
`count_partial_derangements(+integer,+list,-integer)` - `one`

---

count_distinct_derangements/2

Counts the number of distinct derangements of a list (deduplicating equal-valued derangements).

Compilation flags:
```
static
```

Template:
```
count_distinct_derangements(List,Count)
```
Mode and number of proofs:
```
count_distinct_derangements(+list,-integer) - one
```

---

nth_distinct_derangement/3

Returns the distinct derangement at a given zero-based index in default generation order.

Compilation flags:
```
static
```

Template:
```
nth_distinct_derangement(List,Index,Derangement)
```
Mode and number of proofs:
```
nth_distinct_derangement(+list,+integer,-list) - zero_or_one
```

---

distinct_derangement_index/3

Returns the zero-based index of a distinct derangement in default generation order.

Compilation flags:
```
static
```

Template:
```
distinct_derangement_index(List,Derangement,Index)
```
Mode and number of proofs:
```
distinct_derangement_index(+list,+list,-integer) - zero_or_one
```

---

random_derangement/2

Returns a random derangement of a list.

Compilation flags:
`static`

Template:
`random_derangement(List,Derangement)`
Mode and number of proofs:
`random_derangement(+list,-list)` - `zero_or_one`

---

sample_derangements/3

Returns SampleCount random derangements of a list, sampled with replacement.

Compilation flags:
`static`

Template:
`sample_derangements(List,SampleCount,Samples)`
Mode and number of proofs:
`sample_derangements(+list,+integer,-list)` - `zero_or_one`

---

random_distinct_derangement/2

Returns a random distinct derangement of a list (deduplicating equal-valued derangements).

Compilation flags:
`static`

Template:
`random_distinct_derangement(List,Derangement)`
Mode and number of proofs:
`random_distinct_derangement(+list,-list)` - `zero_or_one`

---

sample_distinct_derangements/3

Returns SampleCount random distinct derangements of a list, sampled with replacement after deduplicating equal-valued derangements.

Compilation flags:
```
static
```

Template:
```
sample_distinct_derangements(List,SampleCount,Samples)
```
Mode and number of proofs:
```
sample_distinct_derangements(+list,+integer,-list) - zero_or_one
```

---

next_derangement/3

Returns the next distinct derangement value in lexicographic order induced by the first argument.

Compilation flags:
```
static
```

Template:
```
next_derangement(List,Derangement,Next)
```
Mode and number of proofs:
```
next_derangement(+list,+list,-list) - zero_or_one
```

---

previous_derangement/3

Returns the previous distinct derangement value in lexicographic order induced by the first argument.

Compilation flags:
```
static
```

Template:
```
previous_derangement(List,Derangement,Previous)
```
Mode and number of proofs:
```
previous_derangement(+list,+list,-list) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.69 diagrams

object

1.69.1 `caller_diagram`

Predicates for generating caller diagrams in DOT format.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-12

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *caller_diagram(dot)*

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *predicate/1* *predicate/2* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

xref_diagram, *entity_diagram*

object

1.69.2 `caller_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating caller diagrams showing direct and indirect callers of a predicate or a non-terminal.

Availability:

```
logtalk_load(diagrams(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-12

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *diagram(Format)*

Uses:

list
logtalk
user

Remarks:

(none)

Inherited public predicates:

all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *predicate/2*
  - *predicate/1*
- *Protected predicates*
- *Private predicates*
  - *included_caller_/1*
- *Operators*

Public predicates

predicate/2

Creates a caller diagram for the given predicate or non-terminal using the specified options. Predicates are specified as `Entity::Name/Arity`. Non-terminals are specified as `Entity::Name//Arity`.

Compilation flags:

```
static
```

Template:

```
predicate(QualifiedIndicator,Options)
```

Mode and number of proofs:

```
predicate(+qualified_predicate_indicator,+list(compound)) - one
```

---

predicate/1

Creates a caller diagram for the given predicate using default options. Predicates are specified as `Entity::Name/Arity`. Non-terminals are specified as `Entity::Name//Arity`.

Compilation flags:

```
static
```

Template:

```
predicate(QualifiedIndicator)
```

Mode and number of proofs:

```
predicate(+qualified_predicate_indicator) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`included_caller_/1`

Table of callers already included in the diagram.

Compilation flags:
`dynamic`

Template:
`included_caller_(Caller)`
Mode and number of proofs:
`included_caller_(?qualified_predicate_indicator)` - `zero_or_more`
`included_caller_(?predicate_indicator)` - `zero_or_more`

Operators

(none)

See also

xref_diagram(Format), *entity_diagram(Format)*

object

1.69.3 `cytoscapejs_graph_language`

Predicates for generating diagram files in the Cytoscape Exchange (CX2) JSON format.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-11-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *graph_language_protocol*
Imports:
`public` *options*
Provides:
graph_language_registry::language_object/2

Uses:

list
os
term_io
user

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *edge/6* *file_footer/3* *file_header/3* *graph_footer/5* *graph_header/5* *node/7* *option/2* *option/3* *output_file_name/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - `section_started_/1`
  - `in_nodes_section_/0`
  - `parent_stack_/1`
  - `edge_counter_/1`
  - `node_counter_/1`
  - `node_id_/2`
  - `containment_edge_/2`
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`section_started_/1`

Tracks whether a JSON array section (`nodes` or `edges`) has already emitted at least one element.

Compilation flags:

`dynamic`

Template:

`section_started_(Section)`

Mode and number of proofs:

`section_started_(?atom)` - `zero_or_more`

in_nodes_section_/0

Flag indicating if the exporter is currently writing the nodes section.

Compilation flags:
```
dynamic
```

Mode and number of proofs:
```
in_nodes_section_ - zero_or_one
```

---

parent_stack_/1

Current stack of open graph container identifiers.

Compilation flags:
```
dynamic
```

Template:
```
parent_stack_(Stack)
```
Mode and number of proofs:
```
parent_stack_(?list(nonvar)) - zero_or_one
```

---

edge_counter_/1

Current counter used to generate unique edge identifiers.

Compilation flags:
```
dynamic
```

Template:
```
edge_counter_(Counter)
```
Mode and number of proofs:
```
edge_counter_(?integer) - zero_or_one
```

---

node_counter_/1

Current counter used to generate unique node identifiers.

Compilation flags:
```
dynamic
```

Template:

```
node_counter_(Counter)
```
Mode and number of proofs:
```
node_counter_(?integer) - zero_or_one
```

---

node_id_/2

Maps source node identifiers to generated CX2 numeric identifiers.

Compilation flags:
```
dynamic
```

Template:
```
node_id_(Identifier,NumericId)
```
Mode and number of proofs:
```
node_id_(?nonvar,?integer) - zero_or_more
```

---

containment_edge_/2

Stores pending containment edges between parent and child nodes.

Compilation flags:
```
dynamic
```

Template:
```
containment_edge_(ParentId,ChildId)
```
Mode and number of proofs:
```
containment_edge_(?integer,?integer) - zero_or_more
```

Operators

(none)
object

1.69.4 d2_graph_language

Predicates for generating graph files in the DOT language (version 2.36.0 or later).

Availability:
```
logtalk_load(diagrams(loader))
```

Author: Paulo Moura
Version: 1:6:0
Date: 2026-11-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *graph_language_protocol*
Imports:
`public` *options*
Provides:
graph_language_registry::language_object/2
Uses:
list
os
term_io
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *edge/6* *file_footer/3* *file_header/3* *graph_footer/5* *graph_header/5* *node/7* *option/2* *option/3* *output_file_name/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.69.5 diagram(Format)

- Format - Graph language file format.

Common predicates for generating diagrams.

Availability:
logtalk_load(diagrams(loader))

Author: Paulo Moura
Version: 3:18:0
Date: 2026-03-13

Compilation flags:
static

Extends:
public *options*
Provides:
logtalk::message_prefix_stream/4
logtalk::message_tokens//2
Uses:
graph_language_registry
list
logtalk
modules_diagram_support
os
pairs
type
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *libraries/3*
  - *libraries/2*
  - *libraries/1*
  - *all_libraries/1*
  - *all_libraries/0*
  - *rlibrary/2*
  - *rlibrary/1*
  - *library/2*
  - *library/1*
  - *directories/3*

  - *directories/2*
  - *rdirectory/3*
  - *rdirectory/2*
  - *rdirectory/1*
  - *directory/3*
  - *directory/2*
  - *directory/1*
  - *files/3*
  - *files/2*
  - *files/1*
  - *all_files/1*
  - *all_files/0*
  - *format_object/1*
  - *diagram_description/1*
  - *diagram_name_suffix/1*
- *Protected predicates*
  - *diagram_caption/3*
  - *output_rlibrary/3*
  - *output_library/3*
  - *output_rdirectory/3*
  - *output_externals/1*
  - *output_files/2*
  - *output_file/4*
  - *output_sub_diagrams/1*
  - *reset/0*
  - *output_node/6*
  - *node/6*
  - *edge/5*
  - *output_edges/1*
  - *save_edge/5*
  - *output_missing_externals/1*
  - *not_excluded_file/4*
  - *output_file_path/4*
  - *locate_library/2*
  - *locate_directory/2*
  - *locate_file/5*
  - *ground_entity_identifier/3*
  - *filter_file_extension/3*
  - *filter_external_file_extension/3*
  - *add_link_options/3*
  - *supported_editor_url_scheme_prefix/1*
  - *omit_path_prefix/3*
  - *add_node_zoom_option/4*
  - *message_diagram_description/1*
- *Private predicates*
  - *node_/6*
  - *node_path_/2*
  - *edge_/5*
  - *cycle_edge_/2*
- *Operators*

Public predicates

libraries/3

Creates a diagram for a set of libraries using the specified options. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`libraries(Project,Libraries,Options)`
Mode and number of proofs:
`libraries(+atom,+list(atom),+list(compound))` - `one`

---

libraries/2

Creates a diagram for a set of libraries using the default options. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`libraries(Project,Libraries)`
Mode and number of proofs:
`libraries(+atom,+list(atom))` - `one`

---

libraries/1

Creates a diagram for a set of libraries using the default options. The prefix `libraries` is used for the diagram file name.

Compilation flags:
`static`

Template:
`libraries(Libraries)`
Mode and number of proofs:
`libraries(+list(atom))` - `one`

---

all_libraries/1

Creates a diagram for all loaded libraries using the specified options.

Compilation flags:
`static`

Template:
`all_libraries(Options)`
Mode and number of proofs:
`all_libraries(+list(compound))` - `one`

---

all_libraries/0

Creates a diagram for all loaded libraries using default options.

Compilation flags:
`static`

Mode and number of proofs:
`all_libraries` - `one`

---

rlibrary/2

Creates a diagram for a library and its sub-libraries using the specified options.

Compilation flags:
`static`

Template:
`rlibrary(Library,Options)`
Mode and number of proofs:
`rlibrary(+atom,+list(compound))` - `one`

---

rlibrary/1

Creates a diagram for a library and its sub-libraries using default options.

Compilation flags:
`static`

Template:

```
rlibrary(Library)
```
Mode and number of proofs:
```
rlibrary(+atom) - one
```

---

library/2

Creates a diagram for a library using the specified options.

Compilation flags:
```
static
```

Template:
```
library(Library,Options)
```
Mode and number of proofs:
```
library(+atom,+list(compound)) - one
```

---

library/1

Creates a diagram for a library using default options.

Compilation flags:
```
static
```

Template:
```
library(Library)
```
Mode and number of proofs:
```
library(+atom) - one
```

---

directories/3

Creates a diagram for a set of directories using the specified options. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
```
static
```

Template:
```
directories(Project,Directories,Options)
```
Mode and number of proofs:
```
directories(+atom,+list(atom),+list(compound)) - one
```

---

directories/2

Creates a diagram for a set of directories using the default options. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`directories(Project,Directories)`
Mode and number of proofs:
`directories(+atom,+list(atom))` - `one`

---

rdirectory/3

Creates a diagram for a directory and its sub-directories using the specified options. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`rdirectory(Project,Directory,Options)`
Mode and number of proofs:
`rdirectory(+atom,+atom,+list(compound))` - `one`

---

rdirectory/2

Creates a diagram for a directory and its sub-directories using default options. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`rdirectory(Project,Directory)`
Mode and number of proofs:
`rdirectory(+atom,+atom)` - `one`

---

rdirectory/1

Creates a diagram for a directory and its sub-directories using default options. The name of the directory is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`rdirectory(Directory)`
Mode and number of proofs:
`rdirectory(+atom) - one`

---

directory/3

Creates a diagram for a directory using the specified options. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`directory(Project,Directory,Options)`
Mode and number of proofs:
`directory(+atom,+atom,+list(compound)) - one`

---

directory/2

Creates a diagram for a directory using default options. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`directory(Project,Directory)`
Mode and number of proofs:
`directory(+atom,+atom) - one`

---

directory/1

Creates a diagram for a directory using default options. The name of the directory is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`directory(Directory)`
Mode and number of proofs:
`directory(+atom)` - `one`

---

files/3

Creates a diagram for a set of files using the specified options. The file can be specified by name, basename, full path, or using library notation. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`files(Project,Files,Options)`
Mode and number of proofs:
`files(+atom,+list(atom),+list(compound))` - `one`

---

files/2

Creates a diagram for a set of files using the default options. The file can be specified by name, basename, full path, or using library notation. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`files(Project,Files)`
Mode and number of proofs:
`files(+atom,+list(atom))` - `one`

---

files/1

Creates a diagram for a set of files using the default options. The file can be specified by name, basename, full path, or using library notation. The prefix `files` is used for the diagram file name.

Compilation flags:
`static`

Template:
`files(Files)`
Mode and number of proofs:
`files(+list(atom))` - `one`

---

all_files/1

Creates a diagram for all loaded files using the specified options.

Compilation flags:
`static`

Template:
`all_files(Options)`
Mode and number of proofs:
`all_files(+list(compound))` - `one`

---

all_files/0

Creates a diagram for all loaded files using default options.

Compilation flags:
`static`

Mode and number of proofs:
`all_files` - `one`

---

format_object/1

Returns the identifier of the object implementing the graph language currently being used. Fails if none is specified.

Compilation flags:
`static`

Template:

```
format_object(Object)
```

Mode and number of proofs:

```
format_object(-object_identifier) - zero_or_one
```

---

diagram_description/1

Returns the diagram description.

Compilation flags:

```
static
```

Template:

```
diagram_description(Description)
```

Mode and number of proofs:

```
diagram_description(-atom) - one
```

---

diagram_name_suffix/1

Returns the diagram name suffix.

Compilation flags:

```
static
```

Template:

```
diagram_name_suffix(Suffix)
```

Mode and number of proofs:

```
diagram_name_suffix(-atom) - one
```

Protected predicates

diagram_caption/3

Creates a diagram caption from the diagram description and the subject and its kind.

Compilation flags:

```
static
```

Template:

```
diagram_caption(Kind,Subject,Description)
```

Mode and number of proofs:

```
diagram_caption(+atom,+callable,-atom) - one
```

---

output_rlibrary/3

Generates diagram output for a library and its sub-libraries using the specified options.

Compilation flags:
```
static
```

Template:
```
output_rlibrary(Library,Path,Options)
```
Mode and number of proofs:
```
output_rlibrary(+atom,+atom,+list(compound)) - one
```

---

output_library/3

Generates diagram output for a library using the specified options.

Compilation flags:
```
static
```

Template:
```
output_library(Library,Path,Options)
```
Mode and number of proofs:
```
output_library(+atom,+atom,+list(compound)) - one
```

---

output_rdirectory/3

Generates diagram output for a directory and its sub-directories using the specified options.

Compilation flags:
```
static
```

Template:
```
output_rdirectory(Project,Path,Options)
```
Mode and number of proofs:
```
output_rdirectory(+atom,+atom,+list(compound)) - one
```

---

output_externals/1

Output external nodes using the specified options depending on the value of the boolean option `externals/1`.

Compilation flags:
```
static
```

Template:
```
output_externals(Options)
```
Mode and number of proofs:
```
output_externals(+list(compound)) - one
```

---

output_files/2

Generates diagram output for a list of files using the specified options.

Compilation flags:
```
static
```

Template:
```
output_files(Files,Options)
```
Mode and number of proofs:
```
output_files(+list,+list(compound)) - one
```

---

output_file/4

Generates diagram output for a file using the specified options.

Compilation flags:
```
static
```

Template:
```
output_file(Path,Basename,Directory,Options)
```
Mode and number of proofs:
```
output_file(+atom,+atom,+atom,+list(compound)) - one
```

---

output_sub_diagrams/1

Outputs sub-diagrams using the specified options.

Compilation flags:
```
static
```

Template:
```
output_sub_diagrams(Options)
```
Mode and number of proofs:
```
output_sub_diagrams(+list(compound)) - one
```

---

reset/0

Resets all temporary information used when generating a diagram.

Compilation flags:
```
static
```

Mode and number of proofs:
```
reset - one
```

---

output_node/6

Outputs a graph node.

Compilation flags:
```
static
```

Template:
```
output_node(Identifier,Label,Caption,Contents,Kind,Options)
```
Mode and number of proofs:
```
output_node(+nonvar,+nonvar,+nonvar,+list(nonvar),+atom,+list(compound)) - one
```

---

node/6

Enumerates, by backtracking, all saved nodes.

Compilation flags:
```
static
```

Template:
```
node(Identifier,Label,Caption,Contents,Kind,Options)
```

Mode and number of proofs:

```
node(?nonvar,?nonvar,?nonvar,?list(compound),?atom,?list(compound)) - zero_or_more
```

---

edge/5

Enumerates, by backtracking, all saved edges.

Compilation flags:

```
static
```

Template:

```
edge(From,To,Labels,Kind,Options)
```

Mode and number of proofs:

```
edge(?nonvar,?nonvar,?list(nonvar),?atom,?list(compound)) - zero_or_more
```

---

output_edges/1

Outputs all edges.

Compilation flags:

```
static
```

Template:

```
output_edges(Options)
```

Mode and number of proofs:

```
output_edges(+list(compound)) - one
```

---

save_edge/5

Saves a graph edge.

Compilation flags:

```
static
```

Template:

```
save_edge(From,To,Labels,Kind,Options)
```

Mode and number of proofs:

```
save_edge(+nonvar,+nonvar,+list(nonvar),+atom,+list(compound)) - one
```

---

output_missing_externals/1

Outputs missing external nodes (usually due to unloaded resources) that are referenced from edges.

Compilation flags:

```
static
```

Template:

```
output_missing_externals(Options)
```

Mode and number of proofs:

```
output_missing_externals(+list(compound)) - one
```

---

not_excluded_file/4

True when the given file is not excluded from the generated output. Excluded files may be specified by full path or by basename and with or without extension. Excluded directories may be listed by full or relative path.

Compilation flags:

```
static
```

Template:

```
not_excluded_file(Path,Basename,ExcludedDirectories,ExcludedFiles)
```

Mode and number of proofs:

```
not_excluded_file(+atom,+atom,+list(atom),+list(atom)) - zero_or_one
```

---

output_file_path/4

Returns the output file path.

Compilation flags:

```
static
```

Template:

```
output_file_path(Name,Options,Format,Path)
```

Mode and number of proofs:

```
output_file_path(+atom,+list(atom),+object_identifier,-atom) - one
```

---

locate_library/2

Locates a library given its name.

Compilation flags:
`static`

Template:
`locate_library(Library,Path)`
Mode and number of proofs:
`locate_library(+atom,-atom)` - `one`

---

locate_directory/2

Locates a directory given its name or full path.

Compilation flags:
`static`

Template:
`locate_directory(Directory,Path)`
Mode and number of proofs:
`locate_directory(+atom,-atom)` - `one`

---

locate_file/5

Locates a file given its name, basename, full path, or library notation representation.

Compilation flags:
`static`

Template:
`locate_file(File,Basename,Extension,Directory,Path)`
Mode and number of proofs:
`locate_file(+atom,+atom,+atom,+atom,-atom)` - `one`

---

ground_entity_identifier/3

Converts an entity identifier to a ground term.

Compilation flags:
```
static
```

Template:
```
ground_entity_identifier(Kind,Identifier,GroundIdentifier)
```
Mode and number of proofs:
```
ground_entity_identifier(+atom,+callable,-callable) - one
```

---

filter_file_extension/3

Filters the file name extension depending on the `file_extensions/1` option.

Compilation flags:
```
static
```

Template:
```
filter_file_extension(Basename,Options,Name)
```
Mode and number of proofs:
```
filter_file_extension(+atom,+list(compound),-atom) - one
```

---

filter_external_file_extension/3

Filters the external file name extension depending on the `file_extensions/1` option.

Compilation flags:
```
static
```

Template:
```
filter_external_file_extension(Path,Options,Name)
```
Mode and number of proofs:
```
filter_external_file_extension(+atom,+list(compound),-atom) - one
```

---

add_link_options/3

Adds `url/1`, `urls/2`, and `tooltip/1` link options (for use by the graph language) based on the specified path to the list of options.

Compilation flags:
`static`

Template:
`add_link_options(Path,Options,LinkingOptions)`
Mode and number of proofs:
`add_link_options(+atom,+list(compound),-list(compound))` - `one`

---

supported_editor_url_scheme_prefix/1

Table of prefixes for text editors that supports a URL scheme to open diagram links.

Compilation flags:
`static`

Template:
`supported_editor_url_scheme_prefix(Prefix)`
Mode and number of proofs:
`supported_editor_url_scheme_prefix(?atom)` - `zero_or_more`

---

omit_path_prefix/3

Removes a prefix from a path, returning the relative path, when using the option `omit_path_prefixes/1`. Used mainly for constructing directory and file node identifiers and captions.

Compilation flags:
`static`

Template:
`omit_path_prefix(Path,Options,Relative)`
Mode and number of proofs:
`omit_path_prefix(+atom,+list(compound),-atom)` - `one`

---

add_node_zoom_option/4

Adds node zoom options when using the zoom option.

Compilation flags:

```
static
```

Template:

```
add_node_zoom_option(Identifier,Suffix,Options,NodeOptions)
```

Mode and number of proofs:

```
add_node_zoom_option(+atom,+atom,+list(compound),-list(compound)) - one
```

---

message_diagram_description/1

Diagram description for progress messages.

Compilation flags:

```
static
```

Template:

```
message_diagram_description(Description)
```

Mode and number of proofs:

```
message_diagram_description(?atom) - one
```

Private predicates

node_/6

Table of saved nodes.

Compilation flags:

```
dynamic
```

Template:

```
node_(Identifier,Label,Caption,Contents,Kind,Options)
```

Mode and number of proofs:

```
node_(?nonvar,?nonvar,?nonvar,?list(compound),?atom,?list(compound)) -
zero_or_more
```

---

node_path_/2

Table of node paths.

Compilation flags:
```
dynamic
```

Template:
```
node_path_(Node,Path)
```
Mode and number of proofs:
```
node_path_(?ground,?list(ground)) - zero_or_more
```

---

edge_/5

Table of saved edges.

Compilation flags:
```
dynamic
```

Template:
```
edge_(From,To,Labels,Kind,Options)
```
Mode and number of proofs:
```
edge_(?nonvar,?nonvar,?list(nonvar),?atom,?list(compound)) - zero_or_more
```

---

cycle_edge_/2

Table of edges that are part of a cycle.

Compilation flags:
```
dynamic
```

Template:
```
cycle_edge_(From,To)
```
Mode and number of proofs:
```
cycle_edge_(?nonvar,?nonvar) - zero_or_more
```

Operators

(none)
object

1.69.6 diagrams

Predicates for generating all supported diagrams for libraries, directories, and files in one step using the DOT format.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2019-04-07

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *diagrams(dot)*

Remarks:
(none)

Inherited public predicates:
all_files/0 all_files/1 all_libraries/0 all_libraries/1 directories/2 directories/3 directory/1 directory/2 directory/3 files/1 files/2 files/3 libraries/1 libraries/2 libraries/3 library/1 library/2 rdirectory/1 rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.69.7 `diagrams(Format)`

- `Format` - Graph language file format.

Predicates for generating all supported diagrams for libraries, directories, or files in one step using the specified format.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:8:0
Date: 2019-06-13

Compilation flags:
`static, context_switching_calls`

Uses:
list
os

Remarks:

- Common options: `title/1`, `date/1`, `output_directory/1`, `relation_labels/1`, `node_type_captions/1`, `exclude_files/1`, `exclude_libraries/1`, `url_prefixes/1`, `omit_path_prefix/1`, `entity_url_suffix_target/2`, and `layout/1`.
- Limitations: Some of the provided predicates only make sense for some types of diagrams. Also, fine tuning may require generating individual diagrams directly instead of as a batch using this utility object.

Inherited public predicates:
(none)

- *Public predicates*
  - *libraries/3*
  - *libraries/2*
  - *libraries/1*
  - *all_libraries/1*
  - *all_libraries/0*
  - *rlibrary/2*
  - *rlibrary/1*
  - *library/2*
  - *library/1*

      - *directories/3*
      - *directories/2*
      - *rdirectory/3*
      - *rdirectory/2*
      - *rdirectory/1*
      - *directory/3*
      - *directory/2*
      - *directory/1*
      - *files/3*
      - *files/2*
      - *files/1*
      - *all_files/1*
      - *all_files/0*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

libraries/3

Creates all supported diagrams for a set of libraries using the specified options. The `Project` argument is used as a prefix for the diagram file names.

Compilation flags:
`static`

Template:
`libraries(Project,Libraries,Options)`
Mode and number of proofs:
`libraries(+atom,+list(atom),+list(compound))` - `one`

---

libraries/2

Creates all supported diagrams for a set of libraries using the default options. The `Project` argument is used as a prefix for the diagram file names.

Compilation flags:
`static`

Template:
`libraries(Project,Libraries)`
Mode and number of proofs:
`libraries(+atom,+list(atom))` - `one`

---

libraries/1

Creates all supported diagrams for a set of libraries using the default options. The prefix `libraries` is used for the diagram file names.

Compilation flags:
`static`

Template:
`libraries(Libraries)`
Mode and number of proofs:
`libraries(+list(atom))` - `one`

---

all_libraries/1

Creates all supported diagrams for all loaded libraries using the specified options.

Compilation flags:
`static`

Template:
`all_libraries(Options)`
Mode and number of proofs:
`all_libraries(+list(compound))` - `one`

---

all_libraries/0

Creates all supported diagrams for all loaded libraries using default options.

Compilation flags:
`static`

Mode and number of proofs:
`all_libraries` - `one`

---

rlibrary/2

Creates all supported diagrams for a library and its sub-libraries using the specified options.

Compilation flags:
`static`

Template:

```
rlibrary(Library,Options)
```
Mode and number of proofs:
```
rlibrary(+atom,+list(compound)) - one
```

---

rlibrary/1

Creates all supported diagrams for a library and its sub-libraries using default options.

Compilation flags:
```
static
```

Template:
```
rlibrary(Library)
```
Mode and number of proofs:
```
rlibrary(+atom) - one
```

---

library/2

Creates all supported diagrams for a library using the specified options.

Compilation flags:
```
static
```

Template:
```
library(Library,Options)
```
Mode and number of proofs:
```
library(+atom,+list(compound)) - one
```

---

library/1

Creates all supported diagrams for a library using default options.

Compilation flags:
```
static
```

Template:
```
library(Library)
```
Mode and number of proofs:
```
library(+atom) - one
```

---

directories/3

Creates all supported diagrams for a set of directories using the specified options. The `Project` argument is used as a prefix for the diagram file names.

Compilation flags:
`static`

Template:
`directories(Project,Directories,Options)`
Mode and number of proofs:
`directories(+atom,+list(atom),+list(compound))` - `one`

---

directories/2

Creates all supported diagrams for a directory using default options. The `Project` argument is used as a prefix for the diagram file names.

Compilation flags:
`static`

Template:
`directories(Project,Directories)`
Mode and number of proofs:
`directories(+atom,+list(atom))` - `one`

---

rdirectory/3

Creates all supported diagrams for a directory and its sub-directories using the specified options. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:
`static`

Template:
`rdirectory(Project,Directory,Options)`
Mode and number of proofs:
`rdirectory(+atom,+atom,+list(compound))` - `one`

---

rdirectory/2

Creates all supported diagrams for a directory and its sub-directories using default options. The `Project` argument is used as a prefix for the diagram file name.

Compilation flags:

```
static
```

Template:

```
rdirectory(Project,Directory)
```

Mode and number of proofs:

```
rdirectory(+atom,+atom) - one
```

---

rdirectory/1

Creates all supported diagrams for a directory and its sub-directories using default options. The name of the directory is used as a prefix for the diagram file name.

Compilation flags:

```
static
```

Template:

```
rdirectory(Directory)
```

Mode and number of proofs:

```
rdirectory(+atom) - one
```

---

directory/3

Creates all supported diagrams for a directory using the specified options. The `Project` argument is used as a prefix for the diagram file names.

Compilation flags:

```
static
```

Template:

```
directory(Project,Directory,Options)
```

Mode and number of proofs:

```
directory(+atom,+atom,+list(compound)) - one
```

---

directory/2

Creates all supported diagrams for a directory using default options. The `Project` argument is used as a prefix for the diagram file names.

Compilation flags:
`static`

Template:
`directory(Project,Directory)`
Mode and number of proofs:
`directory(+atom,+atom) - one`

---

directory/1

Creates all supported diagrams for a directory using default options. The name of the directory is used as a prefix for the diagram file names.

Compilation flags:
`static`

Template:
`directory(Directory)`
Mode and number of proofs:
`directory(+atom) - one`

---

files/3

Creates all supported diagrams for a set of files using the specified options. The file can be specified by name, basename, full path, or using library notation. The Project argument is used as a prefix for the diagram file names.

Compilation flags:
`static`

Template:
`files(Project,Files,Options)`
Mode and number of proofs:
`files(+atom,+list(atom),+list(compound)) - one`

---

files/2

Creates all supported diagrams for a set of files using the default options. The file can be specified by name, basename, full path, or using library notation. The Project argument is used as a prefix for the diagram file names.

Compilation flags:

```
static
```

Template:

```
files(Project,Files)
```

Mode and number of proofs:

```
files(+atom,+list(atom)) - one
```

---

files/1

Creates all supported diagrams for a set of files using the default options. The file can be specified by name, basename, full path, or using library notation. The prefix "files" is used for the diagram file names.

Compilation flags:

```
static
```

Template:

```
files(Files)
```

Mode and number of proofs:

```
files(+list(atom)) - one
```

---

all_files/1

Creates all supported diagrams for all loaded files using the specified options.

Compilation flags:

```
static
```

Template:

```
all_files(Options)
```

Mode and number of proofs:

```
all_files(+list(compound)) - one
```

`all_files/0`

Creates all supported diagrams for all loaded files using default options.

Compilation flags:
`static`

Mode and number of proofs:
`all_files` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.69.8 `directory_dependency_diagram`

Predicates for generating directory dependency diagrams in DOT format.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2019-04-07

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *directory_dependency_diagram(dot)*

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

directory_load_diagram, *file_load_diagram*

object

1.69.9 `directory_dependency_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating directory dependency diagrams. A dependency exists when an entity in one directory makes a reference to an entity in another directory.

Availability:

```
logtalk_load(diagrams(loader))
```

Author: Paulo Moura
Version: 3:2:0
Date: 2026-03-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *directory_diagram(Format)*

Uses:

file_dependency_diagram(Format)
list
logtalk

modules_diagram_support

Remarks:

(none)

Inherited public predicates:

all_files/0 all_files/1 all_libraries/0 all_libraries/1 check_option/1 check_options/1 default_option/1 default_options/1 diagram_description/1 diagram_name_suffix/1 directories/2 directories/3 directory/1 directory/2 directory/3 files/1 files/2 files/3 format_object/1 libraries/1 libraries/2 libraries/3 library/1 library/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *sub_diagram_/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

sub_diagram_/2

Table of directory sub-diagrams to support their generation.

Compilation flags:

`dynamic`

Template:

`sub_diagram_(Project,Directory)`

Mode and number of proofs:

`sub_diagram_(?atom,?atom)` - `zero_or_more`

Operators

(none)

See also

directory_load_diagram(Format), *file_load_diagram(Format)*, *library_load_diagram(Format)*

category

1.69.10 `directory_diagram(Format)`

- `Format` - Graph language file format.

Common predicates for generating directory diagrams.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 1:13:0
Date: 2024-12-04

Compilation flags:
`static`

Extends:
`public` *diagram(Format)*
Uses:
list

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *remember_included_directory/1*
  - *remember_referenced_logtalk_directory/1*
  - *remember_referenced_prolog_directory/1*
- *Private predicates*
  - *included_directory_/1*
  - *referenced_logtalk_directory_/1*
  - *referenced_prolog_directory_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

remember_included_directory/1

Remember included Logtalk directory in the diagram.

Compilation flags:
`static`

Template:
`remember_included_directory(Path)`
Mode and number of proofs:
`remember_included_directory(+atom) - one`

---

remember_referenced_logtalk_directory/1

Remember referenced Logtalk directory in the diagram.

Compilation flags:
`static`

Template:
`remember_referenced_logtalk_directory(Path)`
Mode and number of proofs:
`remember_referenced_logtalk_directory(+atom) - one`

---

remember_referenced_prolog_directory/1

Remember referenced Prolog directory in the diagram.

Compilation flags:
`static`

Template:
`remember_referenced_prolog_directory(Path)`
Mode and number of proofs:
`remember_referenced_prolog_directory(+atom) - one`

Private predicates

included_directory_/1

Table of Logtalk directories already included in the diagram.

Compilation flags:
`dynamic`

Template:
`included_directory_(Path)`
Mode and number of proofs:
`included_directory_(?atom)` - `zero_or_more`

---

referenced_logtalk_directory_/1

Table of referenced Logtalk directories in the diagram.

Compilation flags:
`dynamic`

Template:
`referenced_logtalk_directory_(Path)`
Mode and number of proofs:
`referenced_logtalk_directory_(?atom)` - `zero_or_more`

---

referenced_prolog_directory_/1

Table of referenced Prolog directories in the diagram.

Compilation flags:
`dynamic`

Template:
`referenced_prolog_directory_(Path)`
Mode and number of proofs:
`referenced_prolog_directory_(?atom)` - `zero_or_more`

Operators

(none)
object

1.69.11 directory_load_diagram

Predicates for generating directory loading dependency diagrams in DOT format.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2019-04-07

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *directory_load_diagram(dot)*

Remarks:
(none)

Inherited public predicates:
all_files/0 all_files/1 all_libraries/0 all_libraries/1 check_option/1 check_options/1 default_option/1 default_options/1 diagram_description/1 diagram_name_suffix/1 directories/2 directories/3 directory/1 directory/2 directory/3 files/1 files/2 files/3 format_object/1 libraries/1 libraries/2 libraries/3 library/1 library/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

directory_dependency_diagram, *file_dependency_diagram*

object

1.69.12 `directory_load_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating directory loading dependency diagrams.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 3:0:1
Date: 2024-04-01

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *directory_diagram(Format)*

Uses:
file_dependency_diagram(Format)
file_load_diagram(Format)
list
logtalk
modules_diagram_support

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *sub_diagram_/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

sub_diagram_/2

Table of directory sub-diagrams to support their generation.

Compilation flags:
```
dynamic
```

Template:
```
sub_diagram_(Project,Directory)
```
Mode and number of proofs:
```
sub_diagram_(?atom,?atom) - zero_or_more
```

Operators

(none)

See also

directory_dependency_diagram(Format), *file_dependency_diagram(Format)*, *library_dependency_diagram(Format)*

object

1.69.13 dot_graph_language

Predicates for generating graph files in the DOT language (version 2.36.0 or later).

Availability:
```
logtalk_load(diagrams(loader))
```

Author: Paulo Moura
Version: 3:15:2
Date: 2026-11-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *graph_language_protocol*
Imports:
`public` *options*
Provides:
graph_language_registry::language_object/2
Uses:
list
os
term_io
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *edge/6* *file_footer/3* *file_header/3* *graph_footer/5* *graph_header/5* *node/7* *option/2* *option/3* *output_file_name/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.69.14 entity_diagram

Predicates for generating entity diagrams in DOT format with both inheritance and cross-referencing relation edges.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2026-03-12

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *entity_diagram(dot)*

Remarks:
(none)

Inherited public predicates:
all_files/0 all_files/1 all_libraries/0 all_libraries/1 check_option/1 check_options/1 default_option/1 default_options/1 diagram_description/1 diagram_name_suffix/1 directories/2 directories/3 directory/1 directory/2 directory/3 file/1 file/2 files/1 files/2 files/3 format_object/1 libraries/1 libraries/2 libraries/3 library/1 library/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

inheritance_diagram, *uses_diagram*, *caller_diagram*, *xref_diagram*

object

1.69.15 `entity_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating entity diagrams in the specified format with both inheritance and cross-referencing relation edges.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:63:0
Date: 2026-03-14

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *diagram(Format)*

Uses:
coupling_metric
list
logtalk
modules_diagram_support
user

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *file/2*
  - *file/1*
- *Protected predicates*
- *Private predicates*
  - *included_entity_/1*

- *included_module_/1*
- *referenced_entity_/2*
- *referenced_module_/2*

• *Operators*

Public predicates

file/2

Creates a diagram for all entities in a loaded source file using the specified options. The file can be specified by name, basename, full path, or using library notation.

Compilation flags:

```
static
```

Template:

```
file(File,Options)
```

Mode and number of proofs:

```
file(+atom,+list(compound)) - one
```

---

file/1

Creates a diagram for all entities in a loaded source file using default options. The file can be specified by name, basename, full path, or using library notation.

Compilation flags:

```
static
```

Template:

```
file(File)
```

Mode and number of proofs:

```
file(+atom) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

included_entity_/1

Table of Logtalk entities already included in the diagram.

Compilation flags:
`dynamic`

Template:
`included_entity_(Entity)`
Mode and number of proofs:
`included_entity_(?entity_identifier)` - `zero_or_more`

---

included_module_/1

Table of Prolog modules already included in the diagram.

Compilation flags:
`dynamic`

Template:
`included_module_(Module)`
Mode and number of proofs:
`included_module_(?module_identifier)` - `zero_or_more`

---

referenced_entity_/2

Table of referenced Logtalk entities in the diagram.

Compilation flags:
`dynamic`

Template:
`referenced_entity_(Referencer,Entity)`
Mode and number of proofs:
`referenced_entity_(?entity_identifier,?entity_identifier)` - `zero_or_more`

`referenced_module_/2`

Table of referenced Logtalk entities in the diagram.

Compilation flags:
`dynamic`

Template:
`referenced_module_(Referencer,Entity)`
Mode and number of proofs:
`referenced_module_(?entity_identifier,?module_identifier)` - `zero_or_more`

Operators

(none)

See also

inheritance_diagram(Format), *uses_diagram(Format)*, *caller_diagram(Format)*, *xref_diagram(Format)*, *library_diagram(Format)*

object

1.69.16 `file_dependency_diagram`

Predicates for generating file contents dependency diagrams in DOT format. A dependency exists when an entity in one file makes a reference to an entity in another file.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2019-06-13

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *file_dependency_diagram(dot)*

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

file_load_diagram, *directory_load_diagram*, *library_load_diagram*

object

1.69.17 `file_dependency_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating file contents dependency diagrams. A dependency exists when an entity in one file makes a reference to an entity in another file.

Availability:

```
logtalk_load(diagrams(loader))
```

Author: Paulo Moura
Version: 2:28:3
Date: 2024-04-01

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *file_diagram(Format)*

Uses:

entity_diagram(Format)
list
logtalk

modules_diagram_support
os

Remarks:
(none)

Inherited public predicates:
all_files/0 all_files/1 all_libraries/0 all_libraries/1 check_option/1 check_options/1 default_option/1 default_options/1 diagram_description/1 diagram_name_suffix/1 directories/2 directories/3 directory/1 directory/2 directory/3 files/1 files/2 files/3 format_object/1 libraries/1 libraries/2 libraries/3 library/1 library/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *sub_diagram_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

sub_diagram_/1

Table of file sub-diagrams to support their generation.

Compilation flags:
`dynamic`

Template:
`sub_diagram_(File)`
Mode and number of proofs:
`sub_diagram_(?atom)` - `zero_or_more`

Operators

(none)

See also

file_load_diagram(Format), *directory_load_diagram(Format)*, *library_load_diagram(Format)*

category

1.69.18 `file_diagram(Format)`

- `Format` - Graph language file format.

Common predicates for generating file diagrams.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:14:0
Date: 2024-12-04

Compilation flags:
`static`

Extends:
`public` *diagram(Format)*
Uses:
list

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *remember_included_file/1*
  - *remember_referenced_logtalk_file/1*
  - *remember_referenced_prolog_file/1*
- *Private predicates*
  - *included_file_/1*
  - *referenced_logtalk_file_/1*
  - *referenced_prolog_file_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

remember_included_file/1

Remember included Logtalk file in the diagram.

Compilation flags:
`static`

Template:
`remember_included_file(Path)`
Mode and number of proofs:
`remember_included_file(+atom)` - `one`

---

remember_referenced_logtalk_file/1

Remember referenced Logtalk file in the diagram.

Compilation flags:
`static`

Template:
`remember_referenced_logtalk_file(Path)`
Mode and number of proofs:
`remember_referenced_logtalk_file(+atom)` - `one`

---

remember_referenced_prolog_file/1

Remember referenced Prolog file in the diagram.

Compilation flags:
`static`

Template:
`remember_referenced_prolog_file(Path)`
Mode and number of proofs:
`remember_referenced_prolog_file(+atom)` - `one`

Private predicates

included_file_/1

Table of Logtalk files already included in the diagram.

Compilation flags:
`dynamic`

Template:
`included_file_(Path)`
Mode and number of proofs:
`included_file_(?atom)` - `zero_or_more`

---

referenced_logtalk_file_/1

Table of referenced Logtalk files in the diagram.

Compilation flags:
`dynamic`

Template:
`referenced_logtalk_file_(Path)`
Mode and number of proofs:
`referenced_logtalk_file_(?atom)` - `zero_or_more`

---

referenced_prolog_file_/1

Table of referenced Prolog files in the diagram.

Compilation flags:
`dynamic`

Template:
`referenced_prolog_file_(Path)`
Mode and number of proofs:
`referenced_prolog_file_(?atom)` - `zero_or_more`

Operators

(none)
object

1.69.19 `file_load_diagram`

Predicates for generating file loading dependency diagrams in DOT format. A dependency exists when a file loads or includes another file.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2019-06-13

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *file_load_diagram(dot)*

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

file_dependency_diagram, *directory_dependency_diagram*, *library_dependency_diagram*

object

1.69.20 `file_load_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating file loading dependency diagrams. A dependency exists when a file loads or includes another file.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:30:3
Date: 2024-12-05

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *file_diagram(Format)*

Uses:
entity_diagram(Format)
list
logtalk
modules_diagram_support
os

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *sub_diagram_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

sub_diagram_/1

Table of file sub-diagrams to support their generation.

Compilation flags:
`dynamic`

Template:
`sub_diagram_(File)`
Mode and number of proofs:
`sub_diagram_(?atom)` - `zero_or_more`

Operators

(none)

See also

file_dependency_diagram(Format), *directory_dependency_diagram(Format)*, *library_dependency_diagram(Format)*

protocol

1.69.21 graph_language_protocol

Predicates for generating graph files.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2014-12-30

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *output_file_name/2*
  - *file_header/3*
  - *file_footer/3*
  - *graph_header/5*
  - *graph_footer/5*
  - *node/7*
  - *edge/6*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

output_file_name/2

Constructs the diagram file basename by adding a graph language dependent extension to the given name.

Compilation flags:
`static`

Template:
`output_file_name(Name,Basename)`

Mode and number of proofs:
`output_file_name(+atom,-atom) - one`

file_header/3

Writes the output file header using the specified options.

Compilation flags:
```
static
```

Template:
```
file_header(Stream,Identifier,Options)
```
Mode and number of proofs:
```
file_header(+stream_or_alias,+atom,+list(compound)) - one
```

---

file_footer/3

Writes the output file footer using the specified options.

Compilation flags:
```
static
```

Template:
```
file_footer(Stream,Identifier,Options)
```
Mode and number of proofs:
```
file_footer(+stream_or_alias,+atom,+list(compound)) - one
```

---

graph_header/5

Writes a graph header using the specified options.

Compilation flags:
```
static
```

Template:
```
graph_header(Stream,Identifier,Label,Kind,Options)
```
Mode and number of proofs:
```
graph_header(+stream_or_alias,+atom,+atom,+atom,+list(compound)) - one
```

---

graph_footer/5

Writes a graph footer using the specified options.

Compilation flags:

```
static
```

Template:

```
graph_footer(Stream,Identifier,Label,Kind,Options)
```

Mode and number of proofs:

```
graph_footer(+stream_or_alias,+atom,+atom,+atom,+list(compound)) - one
```

---

node/7

Writes a node using the specified options.

Compilation flags:

```
static
```

Template:

```
node(Stream,Identifier,Label,Caption,Lines,Kind,Options)
```

Mode and number of proofs:

```
node(+stream_or_alias,+nonvar,+nonvar,+nonvar,+list(nonvar),+atom,
+list(compound)) - one
```

---

edge/6

Writes an edge between two nodes using the specified options.

Compilation flags:

```
static
```

Template:

```
edge(Stream,Start,End,Labels,Kind,Options)
```

Mode and number of proofs:

```
edge(+stream_or_alias,+nonvar,+nonvar,+list(nonvar),+atom,+list(compound)) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.69.22 `graph_language_registry`

Registry of implemented graph languages.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2020-03-25

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *language_object/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

language_object/2

Table of defined graph languages and their implementation objects.

Compilation flags:
```
static, multifile
```

Template:
```
language_object(Language,Object)
```
Mode and number of proofs:
```
language_object(?atom,?object_identifier) - zero_or_more
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.69.23 inheritance_diagram

Predicates for generating entity diagrams in DOT format with inheritance relation edges but no cross-referencing relation edges.

Availability:
```
logtalk_load(diagrams(loader))
```

Author: Paulo Moura
Version: 2:0:0
Date: 2014-01-15

Compilation flags:
```
static, context_switching_calls
```

Extends:
```
public inheritance_diagram(dot)
```

Remarks:
(none)

Inherited public predicates:

all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *file/1* *file/2* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

entity_diagram, *uses_diagram*, *xref_diagram*

object

1.69.24 `inheritance_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating entity diagrams in the specified format with inheritance relation edges but no cross-referencing relation edges.

Availability:

```
logtalk_load(diagrams(loader))
```

Author: Paulo Moura
Version: 2:20:0
Date: 2024-03-20

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *entity_diagram(Format)*

Uses:
logtalk

Remarks:
(none)

Inherited public predicates:
all_files/0 all_files/1 all_libraries/0 all_libraries/1 check_option/1 check_options/1 default_option/1 default_options/1 diagram_description/1 diagram_name_suffix/1 directories/2 directories/3 directory/1 directory/2 directory/3 file/1 file/2 files/1 files/2 files/3 format_object/1 libraries/1 libraries/2 libraries/3 library/1 library/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

entity_diagram(Format), *uses_diagram(Format)*, *xref_diagram(Format)*

object

1.69.25 `library_dependency_diagram`

Predicates for generating library dependency diagrams in DOT format.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2019-06-13

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public library_dependency_diagram(dot)
```

Remarks:

(none)

Inherited public predicates:

all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

library_load_diagram, *file_load_diagram*, *entity_diagram*

object

1.69.26 `library_dependency_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating library dependency diagrams. A dependency exists when an entity in one library makes a reference to an entity in another library.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:35:0
Date: 2026-03-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *library_diagram(Format)*

Uses:
entity_diagram(Format)
list
logtalk
modules_diagram_support

Remarks:
(none)

Inherited public predicates:
all_files/0 all_files/1 all_libraries/0 all_libraries/1 check_option/1 check_options/1 default_option/1 default_options/1 diagram_description/1 diagram_name_suffix/1 directories/2 directories/3 directory/1 directory/2 directory/3 files/1 files/2 files/3 format_object/1 libraries/1 libraries/2 libraries/3 library/1 library/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *sub_diagram_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`sub_diagram_/1`

Table of library sub-diagrams to support their generation.

Compilation flags:
`dynamic`

Template:
`sub_diagram_(Library)`
Mode and number of proofs:
`sub_diagram_(?atom)` - `zero_or_more`

Operators

(none)

See also

library_load_diagram(Format), *directory_load_diagram(Format)*, *file_load_diagram(Format)*, *entity_diagram(Format)*

category

1.69.27 `library_diagram(Format)`

- `Format` - Graph language file format.

Common predicates for generating library diagrams.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:17:0
Date: 2024-12-04

Compilation flags:
`static`

Extends:
`public` *diagram(Format)*
Uses:
list
user

Remarks:

(none)

Inherited public predicates:

all_files/0 all_files/1 all_libraries/0 all_libraries/1 check_option/1 check_options/1 default_option/1 default_options/1 diagram_description/1 diagram_name_suffix/1 directories/2 directories/3 directory/1 directory/2 directory/3 files/1 files/2 files/3 format_object/1 libraries/1 libraries/2 libraries/3 library/1 library/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
  - *add_library_documentation_url/4*
  - *remember_included_library/2*
  - *remember_referenced_logtalk_library/2*
  - *remember_referenced_prolog_library/2*
- *Private predicates*
  - *included_library_/2*
  - *referenced_logtalk_library_/2*
  - *referenced_prolog_library_/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

add_library_documentation_url/4

Adds a documentation URL when using the option url_prefixes/2.

Compilation flags:

```
static
```

Template:

```
add_library_documentation_url(Kind,Options,Library,NodeOptions)
```

Mode and number of proofs:

```
add_library_documentation_url(+atom,+list(compound),+atom,-list(compound)) - one
```

remember_included_library/2

Remember included Logtalk library in the diagram.

Compilation flags:
`static`

Template:
`remember_included_library(Library,Path)`
Mode and number of proofs:
`remember_included_library(+atom,+atom)` - `one`

---

remember_referenced_logtalk_library/2

Remember referenced Logtalk library in the diagram.

Compilation flags:
`static`

Template:
`remember_referenced_logtalk_library(Library,Path)`
Mode and number of proofs:
`remember_referenced_logtalk_library(+atom,+atom)` - `one`

---

remember_referenced_prolog_library/2

Remember referenced Prolog library in the diagram.

Compilation flags:
`static`

Template:
`remember_referenced_prolog_library(Library,Path)`
Mode and number of proofs:
`remember_referenced_prolog_library(+atom,+atom)` - `one`

Private predicates

included_library_/2

Table of Logtalk libraries already included in the diagram.

Compilation flags:
dynamic

Template:
included_library_(Library,Path)
Mode and number of proofs:
included_library_(?atom,?atom) - zero_or_more

---

referenced_logtalk_library_/2

Table of referenced Logtalk libraries in the diagram.

Compilation flags:
dynamic

Template:
referenced_logtalk_library_(Library,Path)
Mode and number of proofs:
referenced_logtalk_library_(?atom,?atom) - zero_or_more

---

referenced_prolog_library_/2

Table of referenced Prolog libraries in the diagram.

Compilation flags:
dynamic

Template:
referenced_prolog_library_(Library,Path)
Mode and number of proofs:
referenced_prolog_library_(?atom,?atom) - zero_or_more

Operators

(none)

See also

inheritance_diagram(Format), *uses_diagram(Format)*, *xref_diagram(Format)*, *entity_diagram(Format)*

object

1.69.28 `library_load_diagram`

Predicates for generating library loading dependency diagrams in DOT format.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2019-06-13

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *library_load_diagram(dot)*

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

library_dependency_diagram, *file_dependency_diagram*, *entity_diagram*

object

1.69.29 `library_load_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating library loading dependency diagrams.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:33:1
Date: 2024-04-01

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *library_diagram(Format)*

Uses:
entity_diagram(Format)
list
logtalk
modules_diagram_support

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *sub_diagram_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

sub_diagram_/1

Table of library sub-diagrams to support their generation.

Compilation flags:
`dynamic`

Template:
`sub_diagram_(Library)`

Mode and number of proofs:
`sub_diagram_(?atom)` - `zero_or_more`

Operators

(none)

See also

library_dependency_diagram(Format), *directory_dependency_diagram(Format)*, *file_dependency_diagram(Format)*, *entity_diagram(Format)*

object

1.69.30 mermaid_graph_language

Predicates for generating graph files using Mermaid. Requires Mermaid 12 or later.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 0:7:0
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *graph_language_protocol*
Imports:
`public` *options*
Provides:
graph_language_registry::language_object/2
Uses:
list
os
term_io
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *edge/6* *file_footer/3* *file_header/3* *graph_footer/5* *graph_header/5* *node/7* *option/2* *option/3* *output_file_name/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - `edge_index_/1`
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

edge_index_/1

Current edge index counter.

Compilation flags:
`dynamic`

Template:
`edge_index_(Index)`
Mode and number of proofs:
`edge_index_(?integer)` - `zero_or_one`

Operators

(none)
object

1.69.31 modules_diagram_support

Utility predicates for supporting Prolog modules in diagrams.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 0:19:5
Date: 2022-07-08

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Supported backend Prolog systems: ECLiPSe, SICStus Prolog, SWI-Prolog, and YAP.

Inherited public predicates:

(none)

- *Public predicates*
  - *module_property/2*
  - *loaded_file_property/2*
  - *source_file_extension/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

module_property/2

Access to module properties, at least `exports/1`, `file/1`, and `file/2` but also `declares/2`, `defines/2`, `calls/2`, and `provides/3` when possible.

Compilation flags:
`static`

Template:
`module_property(Module,Property)`
Mode and number of proofs:
`module_property(?atom,?callable)` - `zero_or_more`

---

loaded_file_property/2

Access to loaded source file properties, at least `basename/1`, `directory/1` but also `parent/1` when possible.

Compilation flags:
`static`

Template:
`loaded_file_property(File,Property)`
Mode and number of proofs:
`loaded_file_property(?atom,?callable)` - `zero_or_more`

---

source_file_extension/1

Valid source file extension for Prolog source files.

Compilation flags:
`static`

Template:
`source_file_extension(Extension)`
Mode and number of proofs:
`source_file_extension(?atom)` - `one_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.69.32 plantuml_graph_language

Predicates for generating diagrams in PlantUML format.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-11-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *graph_language_protocol*
Imports:
`public` *options*
Provides:
graph_language_registry::language_object/2
Uses:
list
os
term_io

user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *edge/6* *file_footer/3* *file_header/3* *graph_footer/5* *graph_header/5* *node/7* *option/2* *option/3* *output_file_name/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.69.33 `uses_diagram`

Predicates for generating entity diagrams in DOT format with only `uses/2` and `use_module/2` relation edges.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:0:1
Date: 2020-03-27

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *uses_diagram(dot)*

Remarks:

(none)

Inherited public predicates:

all_files/0 all_files/1 all_libraries/0 all_libraries/1 check_option/1 check_options/1 default_option/1 default_options/1 diagram_description/1 diagram_name_suffix/1 directories/2 directories/3 directory/1 directory/2 directory/3 file/1 file/2 files/1 files/2 files/3 format_object/1 libraries/1 libraries/2 libraries/3 library/1 library/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

entity_diagram, *inheritance_diagram*, *xref_diagram*

object

1.69.34 `uses_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating entity diagrams with only `uses/2` and `use_module/2` relation edges.

Availability:

```
logtalk_load(diagrams(loader))
```

Author: Paulo Moura
Version: 2:21:0
Date: 2024-03-20

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *entity_diagram(Format)*

Uses:

logtalk

Remarks:

(none)

Inherited public predicates:

all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *file/1* *file/2* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

entity_diagram(Format), *inheritance_diagram(Format)*, *xref_diagram(Format)*

object

1.69.35 xref_diagram

Predicates for generating predicate call cross-referencing diagrams in DOT format.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2026-03-12

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *xref_diagram(dot)*

Remarks:
(none)

Inherited public predicates:
all_files/0 all_files/1 all_libraries/0 all_libraries/1 check_option/1 check_options/1 default_option/1 default_options/1 diagram_description/1 diagram_name_suffix/1 directories/2 directories/3 directory/1 directory/2 directory/3 entity/1 entity/2 file/1 file/2 files/1 files/2 files/3 format_object/1 libraries/1 libraries/2 libraries/3 library/1 library/2 option/2 option/3 rdirectory/1 rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

caller_diagram, *entity_diagram*, *inheritance_diagram*, *uses_diagram*

object

1.69.36 `xref_diagram(Format)`

- `Format` - Graph language file format.

Predicates for generating predicate call cross-referencing diagrams.

Availability:
`logtalk_load(diagrams(loader))`

Author: Paulo Moura
Version: 2:86:0
Date: 2026-03-12

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *entity_diagram(Format)*
Uses:
atom
list
logtalk
modules_diagram_support
os
user

Remarks:
(none)

Inherited public predicates:
all_files/0 *all_files/1* *all_libraries/0* *all_libraries/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagram_description/1* *diagram_name_suffix/1* *directories/2* *directories/3* *directory/1* *directory/2* *directory/3* *file/1* *file/2* *files/1* *files/2* *files/3* *format_object/1* *libraries/1* *libraries/2* *libraries/3* *library/1* *library/2* *option/2* *option/3* *rdirectory/1* *rdirectory/2* *rdirectory/3* *rlibrary/1* *rlibrary/2* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *entity/2*
  - *entity/1*
- *Protected predicates*
- *Private predicates*

- *included_predicate_/1*
- *referenced_predicate_/1*
- *external_predicate_/1*

- *Operators*

Public predicates

entity/2

Creates a diagram for a single entity using the specified options.

Compilation flags:
`static`

Template:
`entity(Entity,Options)`
Mode and number of proofs:
`entity(+entity_identifier,+list(compound))` - `one`

---

entity/1

Creates a diagram for a single entity using default options.

Compilation flags:
`static`

Template:
`entity(Entity)`
Mode and number of proofs:
`entity(+entity_identifier)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

included_predicate_/1

Table of predicates already included in the diagram for the entity under processing.

Compilation flags:
`dynamic`

Template:

```
included_predicate_(Predicate)
```

Mode and number of proofs:

```
included_predicate_(?predicate_indicator) - zero_or_more
```

`referenced_predicate_/1`

Table of referenced predicates for the entity under processing.

Compilation flags:

```
dynamic
```

Template:

```
referenced_predicate_(Predicate)
```

Mode and number of proofs:

```
referenced_predicate_(?predicate_indicator) - zero_or_more
```

`external_predicate_/1`

Table of external predicate references for all the entities under processing.

Compilation flags:

```
dynamic
```

Template:

```
external_predicate_(Reference)
```

Mode and number of proofs:

```
external_predicate_(?compound) - zero_or_more
```

Operators

(none)

See also

caller_diagram(Format), *entity_diagram(Format)*, *inheritance_diagram(Format)*, *uses_diagram(Format)*

1.70 dictionaries

object

1.70.1 `avltree`

AVL tree implementation of the dictionary protocol. Uses standard order to compare keys.

Availability:
`logtalk_load(dictionaries(loader))`

Author: R.A.O'Keefe, L.Damas, V.S.Costa, Glenn Burgess, Jiri Spitz, and Jan Wielemaker; Logtalk port and additional predicates by Paulo Moura
Version: 1:6:0
Date: 2026-02-10

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *dictionaryp*
Extends:
`public` *term*
Uses:
list

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 apply/4 as_curly_bracketed/2 as_dictionary/2 as_list/2 check/1 clone/3 clone/4 delete/4 delete_max/4 delete_min/4 depth/2 empty/1 ground/1 insert/4 intersection/2 intersection/3 keys/2 lookup/2 lookup/3 lookup/4 map/2 map/3 max/3 min/3 new/1 next/4 numbervars/1 numbervars/3 occurs/2 previous/4 singletons/2 size/2 subsumes/2 subterm/2 update/3 update/4 update/5 valid/1 values/2 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

bintree, *rbtree*

object

1.70.2 `bintree`

Simple binary tree implementation of the dictionary protocol. Uses standard order to compare keys.

Availability:
`logtalk_load(dictionaries(loader))`

Author: Paulo Moura and Paul Fodor
Version: 2:14:0
Date: 2026-09-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *dictionaryp*
Extends:
`public` *term*
Uses:
list

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 apply/4 as_curly_bracketed/2 as_dictionary/2 as_list/2 check/1 clone/3 clone/4 delete/4 delete_max/4 delete_min/4 depth/2 empty/1 ground/1 insert/4 intersection/2 intersection/3 keys/2 lookup/2 lookup/3 lookup/4 map/2 map/3 max/3 min/3 new/1 next/4 numbervars/1 numbervars/3 occurs/2 previous/4 singletons/2 size/2 subsumes/2 subterm/2 update/3 update/4 update/5 valid/1 values/2 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
  - *preorder/2*
  - *inorder/2*
  - *postorder/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

preorder/2

Preorder tree traversal.

Compilation flags:
`static`

Template:
`preorder(Tree,List)`
Mode and number of proofs:
`preorder(@tree,-list)` - `one`

---

inorder/2

Inorder tree traversal.

Compilation flags:
`static`

Template:
`inorder(Tree,List)`
Mode and number of proofs:
`inorder(@tree,-list)` - `one`

---

postorder/2

Postorder tree traversal.

Compilation flags:
`static`

Template:
`postorder(Tree,List)`

Mode and number of proofs:

```
postorder(@tree,-list) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

avltree, *rbtree*, *splaytree*

protocol

1.70.3 `dictionaryp`

Dictionary protocol.

Availability:

```
logtalk_load(dictionaries(loader))
```

Author: Paulo Moura
Version: 2:6:0
Date: 2026-02-10

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *as_dictionary/2*

      - *as_list/2*
      - *as_curly_bracketed/2*
      - *clone/3*
      - *clone/4*
      - *insert/4*
      - *delete/4*
      - *update/4*
      - *update/5*
      - *update/3*
      - *empty/1*
      - *lookup/4*
      - *lookup/3*
      - *lookup/2*
      - *intersection/2*
      - *intersection/3*
      - *previous/4*
      - *next/4*
      - *min/3*
      - *max/3*
      - *delete_min/4*
      - *delete_max/4*
      - *keys/2*
      - *values/2*
      - *map/2*
      - *map/3*
      - *apply/4*
      - *size/2*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

as_dictionary/2

Converts a list of key-value pairs to a dictionary.

Compilation flags:
```
static
```

Template:
```
as_dictionary(Pairs,Dictionary)
```

Mode and number of proofs:
```
as_dictionary(@list(pairs),-dictionary) - one
```

as_list/2

Converts a dictionary to an ordered list (as per standard order) of key-value pairs.

Compilation flags:
```
static
```

Template:
```
as_list(Dictionary,Pairs)
```
Mode and number of proofs:
```
as_list(@dictionary,-list(pairs)) - one
```

---

as_curly_bracketed/2

Creates a curly-bracketed term representation of a dictionary.

Compilation flags:
```
static
```

Template:
```
as_curly_bracketed(Dictionary,Term)
```
Mode and number of proofs:
```
as_curly_bracketed(+dictionary,--term) - one
```

---

clone/3

Clones a dictionary using the same keys but with all values unbound and returning a list of all the pairs in the new clone.

Compilation flags:
```
static
```

Template:
```
clone(Dictionary,Clone,ClonePairs)
```
Mode and number of proofs:
```
clone(+dictionary,-dictionary,-list(pairs)) - one
```

---

clone/4

Clones a dictionary using the same keys but with all values unbound and returning the list of all pairs in the dictionary and in the clone.

Compilation flags:

```
static
```

Template:

```
clone(Dictionary,Pairs,Clone,ClonePairs)
```

Mode and number of proofs:

```
clone(+dictionary,-list(pairs),-dictionary,-list(pairs)) - one
```

---

insert/4

Inserts a key-value pair into a dictionary, returning the updated dictionary. When the key already exists, the associated value is updated.

Compilation flags:

```
static
```

Template:

```
insert(OldDictionary,Key,Value,NewDictionary)
```

Mode and number of proofs:

```
insert(+dictionary,+ground,@term,-dictionary) - one
```

---

delete/4

Deletes a matching key-value pair from a dictionary, returning the updated dictionary. Fails if it cannot find the key or if the key exists but the value does not unify.

Compilation flags:

```
static
```

Template:

```
delete(OldDictionary,Key,Value,NewDictionary)
```

Mode and number of proofs:

```
delete(+dictionary,@ground,?term,-dictionary) - zero_or_one
```

---

update/4

Updates the value associated with Key in a dictionary, returning the updated dictionary. Fails if it cannot find the key.

Compilation flags:

```
static
```

Template:

```
update(OldDictionary,Key,NewValue,NewDictionary)
```

Mode and number of proofs:

```
update(+dictionary,@ground,+term,-dictionary) - zero_or_one
```

---

update/5

Updates the value associated with a key in a dictionary, returning the updated dictionary. Fails if it cannot find the key or if the existing value does not unify.

Compilation flags:

```
static
```

Template:

```
update(OldDictionary,Key,OldValue,NewValue,NewDictionary)
```

Mode and number of proofs:

```
update(+dictionary,@ground,?term,+term,-dictionary) - zero_or_one
```

---

update/3

Updates the key-value pairs in a dictionary, returning the updated dictionary. Fails if it cannot find one of the keys.

Compilation flags:

```
static
```

Template:

```
update(OldDictionary,Pairs,NewDictionary)
```

Mode and number of proofs:

```
update(+dictionary,@list(pair),-dictionary) - zero_or_one
```

---

empty/1

True iff the dictionary is empty.

Compilation flags:
`static`

Template:
`empty(Dictionary)`
Mode and number of proofs:
`empty(@dictionary)` - `zero_or_one`

---

lookup/4

Lookups a matching key-value pair from a dictionary and returns the splayed dictionary with the key at the root. Fails if the key is not found. In implementations that do not update the dictionary on lookup, the same dictionary is returned.

Compilation flags:
`static`

Template:
`lookup(Key,Value,Dictionary,SplayedDictionary)`
Mode and number of proofs:
`lookup(+ground,?term,+tree,-tree)` - `zero_or_one`

---

lookup/3

Lookups a matching key-value pair from a dictionary. Fails if no match is found.

Compilation flags:
`static`

Template:
`lookup(Key,Value,Dictionary)`
Mode and number of proofs:
`lookup(+ground,?term,@dictionary)` - `zero_or_one`
`lookup(-ground,?term,@dictionary)` - `zero_or_more`

---

lookup/2

Lookups all matching key-value pairs from a dictionary. Fails if it cannot find one of the keys or if a value for a key does not unify.

Compilation flags:
`static`

Template:
`lookup(Pairs,Dictionary)`
Mode and number of proofs:
`lookup(+list(pair),@dictionary)` - `zero_or_one`

---

intersection/2

True iff the values of the dictionaries common keys unify. Trivially true when there are no common keys.

Compilation flags:
`static`

Template:
`intersection(Dictionary1,Dictionary2)`
Mode and number of proofs:
`intersection(+dictionary,+dictionary)` - `zero_or_one`

---

intersection/3

Returns the (possibly empty) intersection between two dictionaries when the values of their common keys unify.

Compilation flags:
`static`

Template:
`intersection(Dictionary1,Dictionary2,Intersection)`
Mode and number of proofs:
`intersection(+dictionary,+dictionary,-dictionary)` - `zero_or_one`

---

previous/4

Returns the previous pair in a dictionary given a key. Fails if there is no previous pair.

Compilation flags:
`static`

Template:
`previous(Dictionary,Key,Previous,Value)`
Mode and number of proofs:
`previous(+dictionary,+key,-key,-value)` - `zero_or_one`

---

next/4

Returns the next pair in a dictionary given a key. Fails if there is no next pair.

Compilation flags:
`static`

Template:
`next(Dictionary,Key,Next,Value)`
Mode and number of proofs:
`next(+dictionary,+key,-key,-value)` - `zero_or_one`

---

min/3

Returns the pair with the minimum key (as per standard order) in a dictionary. Fails if the dictionary is empty.

Compilation flags:
`static`

Template:
`min(Dictionary,Key,Value)`
Mode and number of proofs:
`min(+dictionary,-key,-value)` - `zero_or_one`

---

max/3

Returns the pair with the maximum key (as per standard order) in a dictionary. Fails if the dictionary is empty.

Compilation flags:

```
static
```

Template:

```
max(Dictionary,Key,Value)
```

Mode and number of proofs:

```
max(+dictionary,-key,-value) - zero_or_one
```

---

delete_min/4

Deletes the pair with the minimum key (as per standard order) from a dictionary, returning the deleted pair and the updated dictionary. Fails if the dictionary is empty.

Compilation flags:

```
static
```

Template:

```
delete_min(OldDictionary,Key,Value,NewDictionary)
```

Mode and number of proofs:

```
delete_min(+dictionary,-key,-value,-dictionary) - zero_or_one
```

---

delete_max/4

Deletes the pair with the maximum key (as per standard order) from a dictionary, returning the deleted pair and the updated dictionary. Fails if the dictionary is empty.

Compilation flags:

```
static
```

Template:

```
delete_max(OldDictionary,Key,Value,NewDictionary)
```

Mode and number of proofs:

```
delete_max(+dictionary,-key,-value,-dictionary) - zero_or_one
```

---

keys/2

Returns a list with all the dictionary keys in ascending order (as per standard order).

Compilation flags:
```
static
```

Template:
```
keys(Dictionary,Keys)
```
Mode and number of proofs:
```
keys(@dictionary,-list) - one
```

---

values/2

Returns a list with all the dictionary values in ascending order of the keys (as per standard order).

Compilation flags:
```
static
```

Template:
```
values(Dictionary,Values)
```
Mode and number of proofs:
```
values(@dictionary,-list) - one
```

---

map/2

Maps a closure over each dictionary key-value pair. Fails if the mapped closure attempts to modify the keys.

Compilation flags:
```
static
```

Template:
```
map(Closure,Dictionary)
```
Meta-predicate template:
```
map(1,*)
```
Mode and number of proofs:
```
map(@callable,+dictionary) - zero_or_more
```

---

map/3

Maps a closure over each dictionary key-value pair, returning the new dictionary. Fails if the mapped closure attempts to modify the keys.

Compilation flags:
`static`

Template:
`map(Closure,OldDictionary,NewDictionary)`
Meta-predicate template:
`map(2,*,*)`
Mode and number of proofs:
`map(@callable,+dictionary,-dictionary)` - `zero_or_more`

---

apply/4

Applies a closure to a specific key-value pair, returning the new dictionary. Fails if the key cannot be found or if the mapped closure attempts to modify the key.

Compilation flags:
`static`

Template:
`apply(Closure,OldDictionary,Key,NewDictionary)`
Meta-predicate template:
`apply(2,*,*,*)`
Mode and number of proofs:
`apply(+callable,+dictionary,+key,-dictionary)` - `zero_or_one`

---

size/2

Number of dictionary entries.

Compilation flags:
`static`

Template:
`size(Dictionary,Size)`
Mode and number of proofs:
`size(@dictionary,?integer)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

avltree, *bintree*, *rbtree*, *splaytree*

object

1.70.4 `rbtree`

Red-Black tree implementation of the dictionary protocol. Uses standard order to compare keys.

Availability:
`logtalk_load(dictionaries(loader))`

Author: Vitor Santos Costa; Logtalk port and additional predicates by Paulo Moura.
Version: 1:11:0
Date: 2026-02-10

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *dictionaryp*
Extends:
`public` *term*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 apply/4 as_curly_bracketed/2 as_dictionary/2 as_list/2 check/1 clone/3 clone/4 delete/4 delete_max/4 delete_min/4 depth/2 empty/1 ground/1 insert/4 intersection/2 intersection/3 keys/2 lookup/2 lookup/3 lookup/4 map/2 map/3 max/3 min/3 new/1 next/4 numbervars/1 numbervars/3 occurs/2 previous/4 singletons/2 size/2 subsumes/2 subterm/2 update/3 update/4 update/5 valid/1 values/2 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
  - *partial_map/4*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

`partial_map/4`

Applies a closure to the tree pairs identified by a set of keys.

Compilation flags:
`static`

Template:
`partial_map(Tree,Keys,Closure,NewTree)`
Meta-predicate template:
`partial_map(*,*,2,*)`
Mode and number of proofs:
`partial_map(+tree,+list,@closure,-tree)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

avltree, *splaytree*, *bintree*

object

1.70.5 `splaytree`

Splay tree implementation of the dictionary protocol. A splay tree is a self-adjusting binary search tree with the property that recently accessed elements are quick to access again. Uses standard order to compare keys.

Availability:
`logtalk_load(dictionaries(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-10

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *dictionaryp*
Extends:
`public` *term*
Uses:
list

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 apply/4 as_curly_bracketed/2 as_dictionary/2 as_list/2 check/1 clone/3 clone/4 delete/4 delete_max/4 delete_min/4 depth/2 empty/1 ground/1 insert/4 intersection/2 intersection/3 keys/2 lookup/2 lookup/3 lookup/4 map/2 map/3 max/3 min/3 new/1 next/4 numbervars/1 numbervars/3 occurs/2 previous/4 singletons/2 size/2 subsumes/2 subterm/2 update/3 update/4 update/5 valid/1 values/2 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

avltree, *bintree*, *rbtree*

object

1.70.6 two3tree

2-3 tree implementation.

Availability:
`logtalk_load(dictionaries(loader))`

Author: Michael T. Richter
Version: 1:0:1
Date: 2026-06-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *dictionaryp*
Extends:
`public` *term*
Uses:
list

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 apply/4 as_curly_bracketed/2 as_dictionary/2 as_list/2 check/1 clone/3 clone/4 delete/4 delete_max/4 delete_min/4 depth/2 empty/1 ground/1 insert/4 intersection/2 intersection/3 keys/2 lookup/2 lookup/3 lookup/4 map/2 map/3 max/3 min/3 new/1 next/4 numbervars/1 numbervars/3 occurs/2 previous/4 singletons/2 size/2 subsumes/2 subterm/2 update/3 update/4 update/5 valid/1 values/2 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
  - *union/3*
- *Protected predicates*
- *Private predicates*
  - *union_impl/3*
  - *union_impl_list/3*
  - *clone_impl/3*
  - *clone_impl2/4*
  - *handle_root_underflow/2*
  - *delete_impl/5*
  - *fix_node2_left_underflow/6*
  - *fix_node2_right_underflow/6*
  - *fix_node3_left_underflow/9*
  - *fix_node3_middle_underflow/9*
  - *fix_node3_right_underflow/9*
  - *as_curly_bracketed_impl/2*
  - *as_curly_bracketed_impl/3*

    - *insert_impl/4*
    - *insert_empty/4*
    - *insert_node2_2/4*
    - *insert_node2_4/4*
    - *insert_node3_4/4*
    - *insert_node3_7/4*
    - *intersection_impl/4*
    - *is_node4/1*
    - *map_impl/2*
    - *map_impl/3*
    - *next_impl/5*
    - *previous_impl/5*
    - *node2_from_node4/2*
    - *update_impl/3*
    - *collect/4*
    - *select/4*
- *Operators*

Public predicates

union/3

Computes the union of two dictionaries. If a key appears in both, the value from the larger dictionary is kept.

Compilation flags:
`static`

Template:
`union(Tree1,Tree2,Union)`

Mode and number of proofs:
`union(+two3tree,+two3tree,-two3tree)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

union_impl/3

Inserts all key-value pairs of the smaller tree into the larger tree to compute the union.

Compilation flags:
`static`

Template:
`union_impl(LargerTree,SmallerTree,Union)`

Mode and number of proofs:
`union_impl(+two3tree,+two3tree,-two3tree)` - `zero_or_one`

---

union_impl_list/3

Inserts a list of key-value pairs into a dictionary one by one.

Compilation flags:
`static`

Template:
`union_impl_list(Pairs,Dictionary,Union)`
Mode and number of proofs:
`union_impl_list(+list(pair),+two3tree,-two3tree)` - `zero_or_one`

---

clone_impl/3

Recursively clones a tree, leaving all values unbound and collecting the pairs of the clone.

Compilation flags:
`static`

Template:
`clone_impl(Tree,Clone,ClonePairs)`
Mode and number of proofs:
`clone_impl(+two3tree,-two3tree,-list(pair))` - `one`

---

clone_impl2/4

Recursively clones a tree, returning both the original pairs and the clone pairs.

Compilation flags:
`static`

Template:
`clone_impl2(Tree,Clone,Pairs,ClonePairs)`
Mode and number of proofs:
`clone_impl2(+two3tree,-two3tree,-list(pair),-list(pair))` - `one`

---

handle_root_underflow/2

After a deletion, if the root has become a 2-node with a single child, replace it by that child; otherwise keep the root.

Compilation flags:
```
static
```

Template:
```
handle_root_underflow(Tree,NewTree)
```
Mode and number of proofs:
```
handle_root_underflow(+term,-term) - one
```

---

delete_impl/5

Recursive deletion that returns the resulting tree and a flag indicating whether an underflow has occurred at the current node.

Compilation flags:
```
static
```

Template:
```
delete_impl(Tree,Key,Value,NewTree,Underflow)
```
Mode and number of proofs:
```
delete_impl(+term,+term,?term,-term,-boolean) - zero_or_one
```

---

fix_node2_left_underflow/6

Repairs an underflow after a deletion in the left subtree of a node2.

Compilation flags:
```
static
```

Template:
```
fix_node2_left_underflow(LeftSubtree,RightSubtree,Key,Value,NewTree,Underflow)
```
Mode and number of proofs:
```
fix_node2_left_underflow(+term,+term,+term,+term,-term,-boolean) - one
```

---

fix_node2_right_underflow/6

Repairs an underflow after a deletion in the right subtree of a node2.

Compilation flags:

```
static
```

Template:

```
fix_node2_right_underflow(LeftSubtree,RightSubtree,Key,Value,NewTree,Underflow)
```

Mode and number of proofs:

```
fix_node2_right_underflow(+term,+term,+term,+term,-term,-boolean) - one
```

---

fix_node3_left_underflow/9

Repairs an underflow after a deletion in the left subtree of a node3.

Compilation flags:

```
static
```

Template:

```
fix_node3_left_underflow(LeftSubtree,MiddleSubtree,RightSubtree,LeftKey,
LeftValue,RightKey,RightValue,NewTree,Underflow)
```

Mode and number of proofs:

```
fix_node3_left_underflow(+term,+term,+term,+term,+term,+term,+term,-term,
-boolean) - one
```

---

fix_node3_middle_underflow/9

Repairs an underflow after a deletion in the middle subtree of a node3.

Compilation flags:

```
static
```

Template:

```
fix_node3_middle_underflow(LeftSubtree,MiddleSubtree,RightSubtree,LeftKey,
LeftValue,RightKey,RightValue,NewTree,Underflow)
```

Mode and number of proofs:

```
fix_node3_middle_underflow(+term,+term,+term,+term,+term,+term,+term,-term,
-boolean) - one
```

---

fix_node3_right_underflow/9

Repairs an underflow after a deletion in the right subtree of a node3.

Compilation flags:

```
static
```

Template:

```
fix_node3_right_underflow(LeftSubtree,MiddleSubtree,RightSubtree,LeftKey,
LeftValue,RightKey,RightValue,NewTree,Underflow)
```

Mode and number of proofs:

```
fix_node3_right_underflow(+term,+term,+term,+term,+term,+term,+term,-term,
-boolean) - one
```

---

as_curly_bracketed_impl/2

Builds a curly-bracketed term from a list of key-value pairs.

Compilation flags:

```
static
```

Template:

```
as_curly_bracketed_impl(Pairs,Term)
```

Mode and number of proofs:

```
as_curly_bracketed_impl(+list(pairs),-term) - one
```

---

as_curly_bracketed_impl/3

Helper that accumulates a comma-separated list inside curly braces.

Compilation flags:

```
static
```

Template:

```
as_curly_bracketed_impl(Pairs,Pair,Term)
```

Mode and number of proofs:

```
as_curly_bracketed_impl(+list(pairs),+pair,-term) - one
```

---

insert_impl/4

Recursive insertion that may return a node4 on the way up.

Compilation flags:
```
static
```

Template:
```
insert_impl(OldTree,Key,Value,NewTree)
```
Mode and number of proofs:
```
insert_impl(+term,+term,+term,-term) - one
```

---

insert_empty/4

Inserts into an empty tree.

Compilation flags:
```
static
```

Template:
```
insert_empty(EmptyTree,Key,Value,NewTree)
```
Mode and number of proofs:
```
insert_empty(+empty,+term,+term,-term) - one
```

---

insert_node2_2/4

Inserts a key-value pair into a leaf 2-node, creating a leaf 3-node.

Compilation flags:
```
static
```

Template:
```
insert_node2_2(Tree,Key,Value,NewTree)
```
Mode and number of proofs:
```
insert_node2_2(+term,+term,+term,-term) - one
```

---

insert_node2_4/4

Inserts into a non-leaf 2-node, possibly splitting a child 4-node.

Compilation flags:
```
static
```

Template:
```
insert_node2_4(Tree,Key,Value,NewTree)
```
Mode and number of proofs:
```
insert_node2_4(+term,+term,+term,-term) - one
```

---

insert_node3_4/4

Inserts a key-value pair into a leaf 3-node, creating a leaf 4-node.

Compilation flags:
```
static
```

Template:
```
insert_node3_4(Tree,Key,Value,NewTree)
```
Mode and number of proofs:
```
insert_node3_4(+term,+term,+term,-term) - one
```

---

insert_node3_7/4

Inserts into a non-leaf 3-node, possibly splitting a child 4-node.

Compilation flags:
```
static
```

Template:
```
insert_node3_7(Tree,Key,Value,NewTree)
```
Mode and number of proofs:
```
insert_node3_7(+term,+term,+term,-term) - one
```

---

intersection_impl/4

Computes the intersection of a list of key-value pairs with a tree, inserting common pairs into an accumulator.

Compilation flags:
`static`

Template:
`intersection_impl(Pairs,Tree,Accumulator,Intersection)`
Mode and number of proofs:
`intersection_impl(+list(pair),+two3tree,+list(pair),-list(pair))` - `zero_or_one`

---

is_node4/1

True if the term is a 4-node (temporary representation).

Compilation flags:
`static`

Template:
`is_node4(Term)`
Mode and number of proofs:
`is_node4(+term)` - `zero_or_one`

---

map_impl/2

Traverses the tree and applies a closure to each key-value pair.

Compilation flags:
`static`

Template:
`map_impl(Tree,Closure)`
Meta-predicate template:
`map_impl(*,1)`
Mode and number of proofs:
`map_impl(+two3tree,@callable)` - `zero_or_more`

---

map_impl/3

Traverses the tree, applies a closure to each key-value pair, and builds a new tree with the results.

Compilation flags:
`static`

Template:
`map_impl(OldTree,Closure,NewTree)`
Meta-predicate template:
`map_impl(*,2,*)`
Mode and number of proofs:
`map_impl(+two3tree,@callable,-two3tree)` - `zero_or_more`

---

next_impl/5

Recursively finds the next key-value pair after a given key.

Compilation flags:
`static`

Template:
`next_impl(Tree,Key,NextKey,NextValue,Successor)`
Mode and number of proofs:
`next_impl(+two3tree,+key,-key,-value,+term)` - `zero_or_one`

---

previous_impl/5

Recursively finds the previous key-value pair before a given key.

Compilation flags:
`static`

Template:
`previous_impl(Tree,Key,PreviousKey,PreviousValue,Predecessor)`
Mode and number of proofs:
`previous_impl(+two3tree,+key,-key,-value,+term)` - `zero_or_one`

---

node2_from_node4/2

Converts a 4-node (temporary) into a balanced 2-node with two 2-node children.

Compilation flags:

```
static
```

Template:

```
node2_from_node4(Node4,Tree)
```

Mode and number of proofs:

```
node2_from_node4(+term,-term) - one
```

---

update_impl/3

Updates a dictionary with a list of key-value pairs sequentially.

Compilation flags:

```
static
```

Template:

```
update_impl(Pairs,OldTree,NewTree)
```

Mode and number of proofs:

```
update_impl(@list(pair),+two3tree,-two3tree) - zero_or_one
```

---

collect/4

In-order traversal accumulating selected projections of nodes.

Compilation flags:

```
static
```

Template:

```
collect(Tree,Selector,Accumulator,ReversedResult)
```

Mode and number of proofs:

```
collect(+two3tree,+selector,+list(term),-list(term)) - one
```

`select/4`

Projects a key-value pair according to the selector.

Compilation flags:
`static`

Template:
`select(Selector,Key,Value,Element)`
Mode and number of proofs:
`select(+selector,+key,+value,-term) - one`

Operators

(none)

See also

avltree, *bintree*, *rbtree*, *splaytree*, *dictionaryp*

1.71 dif

object

1.71.1 `dif`

Provides dif/2 and derived predicates.

Availability:
`logtalk_load(dif(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2026-10-28

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Supported backend Prolog systems: B-Prolog, ECLiPSe, SICStus Prolog, Shumway Prolog, SWI-Prolog, Trealla Prolog, and YAP.

Inherited public predicates:

(none)

- *Public predicates*
  - *dif/2*
  - *dif/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

dif/2

Sets a constraint that is true iff the two terms are different.

Compilation flags:
`static`

Template:
`dif(Term1,Term2)`
Mode and number of proofs:
`dif(+term,+term)` - `zero_or_one`

---

dif/1

Sets a set of constraints that are true iff all terms in a list are different.

Compilation flags:
`static`

Template:
`dif(Terms)`
Mode and number of proofs:
`dif(+list(term))` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.72 differential_evolution

object

1.72.1 `differential_evolution(Problem)`

- `Problem` - Problem object implementing `differential_evolution_problem_protocol`.

Differential Evolution using the Xoshiro128++ random number generator via `fast_random(xoshiro128pp)`.

Availability:
`logtalk_load(differential_evolution(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *differential_evolution(Problem,fast_random(xoshiro128pp))*

Remarks:
(none)

Inherited public predicates:
check_option/1 check_options/1 default_option/1 default_options/1 option/2 option/3 run/2 run/3 run/4 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

differential_evolution(Problem,Random), *differential_evolution_problem_protocol*

object

1.72.2 `differential_evolution(Problem,Random)`

- `Problem` - Problem object implementing `differential_evolution_problem_protocol`.
- `Random` - Random number generator object providing `between/3`, `random/1`, and `randomize/1` predicates.

Differential Evolution (DE/rand/1/bin, DE/rand/1/exp, DE/best/1/bin, and DE/current-to-best/1/bin) metaheuristic for continuous bounded optimization.

Availability:
`logtalk_load(differential_evolution(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-10-19

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
linear_algebra
list
numberlist
type

Remarks:

- Strategies: The `strategy(rand/1/bin)` (default), `strategy(rand/1/exp)`, `strategy(best/1/bin)`, and `strategy(current-to-best/1/bin)` options select the classic mutation strategies.
- Optimization objective: The `objective(minimize|maximize)` option selects the fitness ordering.

- Boundary handling: Trial vectors that leave the bounds are clamped to the nearest bound.
- Seed control: The `seed(S)` option initializes the random number generator for reproducible runs.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *run/2*
  - *run/3*
  - *run/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

run/2

Runs Differential Evolution with default options and returns the best position and fitness found.

Compilation flags:

`static`

Template:

`run(BestPosition,BestFitness)`

Mode and number of proofs:

`run(-list(number),-number)` - `one_or_error`

Exceptions:

`Bounds` returned by `Problem::position_bounds/1` is a variable:

`instantiation_error`

`Positions` returned by `Problem::initial_positions/1` is a variable:

`instantiation_error`

`Fitness` returned by `Problem::fitness/2` is a variable:

`instantiation_error`

`Bounds` returned by `Problem::position_bounds/1` is not a valid list of position bounds:

`domain_error(position_bounds,Bounds)`

`Positions` returned by `Problem::initial_positions/1` is not a valid initial population:

`domain_error(initial_positions,Positions)`

`Fitness` returned by `Problem::fitness/2` is not a number:

`domain_error(fitness,Fitness)`

run/3

Runs Differential Evolution with the given options and returns the best position and fitness found.

Compilation flags:
`static`

Template:
`run(BestPosition,BestFitness,Options)`
Mode and number of proofs:
`run(-list(number),-number,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
`Bounds` returned by `Problem::position_bounds/1` is a variable:
`instantiation_error`
`Positions` returned by `Problem::initial_positions/1` is a variable:
`instantiation_error`
`Fitness` returned by `Problem::fitness/2` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Bounds` returned by `Problem::position_bounds/1` is not a valid list of position bounds:
`domain_error(position_bounds,Bounds)`
`Positions` returned by `Problem::initial_positions/1` is not a valid initial population:
`domain_error(initial_positions,Positions)`
`Fitness` returned by `Problem::fitness/2` is not a number:
`domain_error(fitness,Fitness)`

---

run/4

Runs Differential Evolution with the given options and returns the best position, best fitness, and run statistics.

Compilation flags:
`static`

Template:
`run(BestPosition,BestFitness,Statistics,Options)`
Mode and number of proofs:
`run(-list(number),-number,-list(compound),+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
`Bounds` returned by `Problem::position_bounds/1` is a variable:
`instantiation_error`
`Positions` returned by `Problem::initial_positions/1` is a variable:
`instantiation_error`
`Fitness` returned by `Problem::fitness/2` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Bounds` returned by `Problem::position_bounds/1` is not a valid list of position bounds:
`domain_error(position_bounds,Bounds)`
`Positions` returned by `Problem::initial_positions/1` is not a valid initial population:
`domain_error(initial_positions,Positions)`
`Fitness` returned by `Problem::fitness/2` is not a number:
`domain_error(fitness,Fitness)`

Remarks:

- Statistics list: Contains `generations(N)`, `evaluations(E)`, `improvements(I)`, `final_mean_fitness(M)`, and `final_diversity(D)`.

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

differential_evolution(Problem), *differential_evolution_problem_protocol*

protocol

1.72.3 differential_evolution_problem_protocol

Protocol for continuous bounded Differential Evolution problem definitions.

Availability:
`logtalk_load(differential_evolution(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-10-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *position_bounds/1*
  - *fitness/2*
  - *initial_positions/1*
  - *stop_condition/3*
  - *progress/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

position_bounds/1

Returns a ground, non-empty proper list of `Lower-Upper` numeric bound pairs, with `Lower =< Upper`. Fixed dimensions are allowed.

Compilation flags:
`static`

Template:
`position_bounds(Bounds)`
Mode and number of proofs:
`position_bounds(-list(pair))` - `one`

fitness/2

Computes the fitness of a candidate solution. The optimization direction is selected by the `objective/1` option.

Compilation flags:
`static`

Template:
`fitness(Position,Fitness)`
Mode and number of proofs:
`fitness(+list(number),-number)` - `one`

---

initial_positions/1

Optional. Returns exactly the configured population size in ground initial positions inside the bounds. When not defined, the algorithm generates a continuous random initial population.

Compilation flags:
`static`

Template:
`initial_positions(Positions)`
Mode and number of proofs:
`initial_positions(-list(list(number)))` - `zero_or_one`

---

stop_condition/3

Optional. True when the search should stop given the completed generation, best position, and best fitness.

Compilation flags:
`static`

Template:
`stop_condition(Generation,BestPosition,BestFitness)`
Mode and number of proofs:
`stop_condition(+non_negative_integer,+list(number),+number)` - `zero_or_one`

`progress/5`

Optional. Called at most `updates(N)` times, including one final report, with generation, best position, best fitness, mean fitness, and mean Euclidean distance from the centroid.

Compilation flags:
`static`

Template:
`progress(Generation,BestPosition,BestFitness,MeanFitness,Diversity)`

Mode and number of proofs:
`progress(+non_negative_integer,+list(number),+number,+number,+number)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

differential_evolution(Problem), *differential_evolution(Problem,Random)*

1.73 dimension_reduction_protocols

category

1.73.1 `dimension_reducer_common`

Shared predicates for dimension reducer learning defaults, dataset helpers, transformation, export, and printing.

Availability:
`logtalk_load(dimension_reduction_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-07

Compilation flags:
`static`

Implements:

public *dimension_reducer_protocol*

Extends:

public *options*

Uses:

format
linear_algebra
list
numberlist
population
type
user

Remarks:

(none)

Inherited public predicates:

check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *check_component_count/3*
  - *dimension_reducer_data/3*
  - *dimension_reducer_diagnostics_data/2*
  - *print_dimension_reducer_properties/1*
  - *example_attribute_values/2*
  - *dataset_attributes/2*
  - *check_continuous_attributes/1*
  - *check_examples_non_empty/2*
  - *check_example_values/2*
  - *check_example_attributes/2*
  - *attribute_value/3*
  - *build_encoders/4*
  - *base_dimension_reducer_diagnostics/6*
  - *preprocessing_diagnostics/3*
  - *iterative_dimension_reducer_diagnostics/11*
  - *component_iteration_diagnostics/4*
  - *zero_vector_like/2*
  - *basis_vector/3*
  - *initial_vectors/2*
  - *basis_initial_vectors/3*
  - *extract_components/5*
  - *known_attribute_values/3*
  - *examples_to_rows/3*
  - *encode_instance/3*
  - *encoder_attribute_names/2*

- *project_components/4*
- *valid_linear_encoders/1*
- *valid_projection_components/2*
- *valid_dimension_reducer_metadata/1*
- *print_dimension_reducer_details/3*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

check_component_count/3

Checks that a requested component count does not exceed the supported maximum and returns the accepted count.

Compilation flags:

```
static
```

Template:

```
check_component_count(RequestedComponentCount,MaxComponentCount,ComponentCount)
```

Mode and number of proofs:

```
check_component_count(+integer,+integer,-integer) - one
```

---

dimension_reducer_data/3

Default hook predicate for exposing the learned encoders and projection components from a reducer term. Importing implementations may override it when using a non-standard reducer representation.

Compilation flags:

```
static
```

Template:

```
dimension_reducer_data(DimensionReducer,Encoders,Components)
```

Mode and number of proofs:

```
dimension_reducer_data(+compound,-list,-list) - one
```

---

dimension_reducer_diagnostics_data/2

Default hook predicate for exposing diagnostics metadata from a reducer term. Importing implementations may override it when using a non-standard reducer representation.

Compilation flags:

```
static
```

Template:

```
dimension_reducer_diagnostics_data(DimensionReducer,Diagnostics)
```

Mode and number of proofs:

```
dimension_reducer_diagnostics_data(+compound,-list(compound)) - one
```

---

print_dimension_reducer_properties/1

Hook predicate that importing dimension reducer implementations must define in order to print the learned reducer in a human-readable form.

Compilation flags:

```
static
```

Template:

```
print_dimension_reducer_properties(DimensionReducer)
```

Mode and number of proofs:

```
print_dimension_reducer_properties(+compound) - one
```

---

example_attribute_values/2

Hook predicate that importing dimension reducer implementations must define in order to extract example attribute values from their local training example representation.

Compilation flags:

```
static
```

Template:

```
example_attribute_values(Example,AttributeValues)
```

Mode and number of proofs:

```
example_attribute_values(+compound,-list(pair)) - one
```

---

dataset_attributes/2

Collects the dataset attribute declarations as *Attribute-Values* pairs.

Compilation flags:
```
static
```

Template:
```
dataset_attributes(Dataset,Attributes)
```
Mode and number of proofs:
```
dataset_attributes(+object_identifier,-list(pair)) - one
```

---

check_continuous_attributes/1

Checks that all declared dataset attributes are continuous.

Compilation flags:
```
static
```

Template:
```
check_continuous_attributes(Attributes)
```
Mode and number of proofs:
```
check_continuous_attributes(+list(pair)) - one
```

---

check_examples_non_empty/2

Checks that a training example collection is not empty.

Compilation flags:
```
static
```

Template:
```
check_examples_non_empty(Dataset,Examples)
```
Mode and number of proofs:
```
check_examples_non_empty(+object_identifier,+list) - one
```

---

check_example_values/2

Checks that all example attribute values are present and numeric for the declared attributes.

Compilation flags:
`static`

Template:
`check_example_values(Examples,AttributeNames)`
Mode and number of proofs:
`check_example_values(+list,+list(atom))` - `one`

---

check_example_attributes/2

Checks that a single example contains numeric values for all declared attributes.

Compilation flags:
`static`

Template:
`check_example_attributes(AttributeNames,AttributeValues)`
Mode and number of proofs:
`check_example_attributes(+list(atom),+list(pair))` - `one`

---

attribute_value/3

Looks up an attribute value in a list of *Attribute-Value* pairs.

Compilation flags:
`static`

Template:
`attribute_value(Attribute,AttributeValues,Value)`
Mode and number of proofs:
`attribute_value(+atom,+list(pair),-term)` - `one`

---

build_encoders/4

Builds continuous feature encoders by computing per-attribute centering and optional scaling statistics.

Compilation flags:

```
static
```

Template:

```
build_encoders(AttributeNames,Examples,Options,Encoders)
```

Mode and number of proofs:

```
build_encoders(+list(atom),+list,+list(compound),-list(compound)) - one
```

---

base_dimension_reducer_diagnostics/6

Builds common diagnostics metadata terms for a learned reducer and appends reducer-specific diagnostics terms.

Compilation flags:

```
static
```

Template:

```
base_dimension_reducer_diagnostics(Model,AttributeNames,Components,Options,
ExtraDiagnostics,Diagnostics)
```

Mode and number of proofs:

```
base_dimension_reducer_diagnostics(+atom,+list(atom),+list,+list(compound),
+list(compound),-list(compound)) - one
```

---

preprocessing_diagnostics/3

Builds shared preprocessing diagnostics metadata from an explicit centering flag and the effective training options.

Compilation flags:

```
static
```

Template:

```
preprocessing_diagnostics(Center,Options,Preprocessing)
```

Mode and number of proofs:

```
preprocessing_diagnostics(+boolean,+list(compound),-list(compound)) - one
```

---

iterative_dimension_reducer_diagnostics/11

Builds diagnostics metadata for reducers that report sample counts, iterative convergence terms, and optional leading or trailing reducer-specific diagnostics.

Compilation flags:

```
static
```

Template:

```
iterative_dimension_reducer_diagnostics(Model,AttributeNames,Components,
SampleCount,Options,LeadingDiagnostics,Convergence,Iterations,FinalDelta,
TrailingDiagnostics,Diagnostics)
```

Mode and number of proofs:

```
iterative_dimension_reducer_diagnostics(+atom,+list(atom),+list,+integer,
+list(compound),+list(compound),+term,+term,+term,+list(compound),
-list(compound)) - one
```

---

component_iteration_diagnostics/4

Extracts per-component convergence, iteration, and final-delta lists from component diagnostics records.

Compilation flags:

```
static
```

Template:

```
component_iteration_diagnostics(ComponentDiagnostics,Convergences,
IterationCounts,FinalDeltas)
```

Mode and number of proofs:

```
component_iteration_diagnostics(+list(compound),-list(atom),-list(integer),
-list(number)) - one
```

---

zero_vector_like/2

Constructs a zero vector matching the length of the first vector in a list of vectors, or returns the empty list when the input is empty.

Compilation flags:

```
static
```

Template:

```
zero_vector_like(Vectors,ZeroVector)
```

Mode and number of proofs:

```
zero_vector_like(+list(list(number)),-list(number)) - one
```

---

basis_vector/3

Constructs a canonical basis vector for the requested size and one-based index.

Compilation flags:
```
static
```

Template:
```
basis_vector(Size,Index,Vector)
```
Mode and number of proofs:
```
basis_vector(+integer,+integer,-list(number)) - one
```

---

initial_vectors/2

Constructs the default all-ones initial vector followed by canonical basis vectors for the requested size.

Compilation flags:
```
static
```

Template:
```
initial_vectors(Size,Vectors)
```
Mode and number of proofs:
```
initial_vectors(+integer,-list(list(number))) - one
```

---

basis_initial_vectors/3

Constructs canonical basis vectors from the given one-based index up to the requested size.

Compilation flags:
```
static
```

Template:
```
basis_initial_vectors(Index,Size,Vectors)
```
Mode and number of proofs:
```
basis_initial_vectors(+integer,+integer,-list(list(number))) - one
```

---

extract_components/5

Extracts leading positive eigen-components from a numeric matrix using the shared symmetric eigensolver until the requested count or the configured tolerance is reached.

Compilation flags:

```
static
```

Template:

```
extract_components(Matrix,Requested,Options,Components,Eigenvalues)
```

Mode and number of proofs:

```
extract_components(+list(list(number)),+integer,+list(compound),
-list(list(number)),-list(number)) - one
```

---

known_attribute_values/3

Collects the known numeric values for a given attribute across the training examples.

Compilation flags:

```
static
```

Template:

```
known_attribute_values(Examples,Attribute,Values)
```

Mode and number of proofs:

```
known_attribute_values(+list,+atom,-list(number)) - one
```

---

examples_to_rows/3

Encodes a list of training examples into numeric feature rows using the importing reducer example hook and learned encoders.

Compilation flags:

```
static
```

Template:

```
examples_to_rows(Examples,Encoders,Rows)
```

Mode and number of proofs:

```
examples_to_rows(+list,+list(compound),-list(list(number))) - one
```

---

encode_instance/3

Encodes an instance using the learned continuous attribute encoders.

Compilation flags:
```
static
```

Template:
```
encode_instance(Encoders,AttributeValues,Features)
```
Mode and number of proofs:
```
encode_instance(+list(compound),+list(pair),-list(number)) - one
```

---

encoder_attribute_names/2

Collects encoder attribute names preserving encoder order.

Compilation flags:
```
static
```

Template:
```
encoder_attribute_names(Encoders,AttributeNames)
```
Mode and number of proofs:
```
encoder_attribute_names(+list(compound),-list(atom)) - one
```

---

project_components/4

Projects encoded features onto the learned components and returns *component_N-Score* pairs.

Compilation flags:
```
static
```

Template:
```
project_components(Components,Features,Index,ReducedInstance)
```
Mode and number of proofs:
```
project_components(+list(list(number)),+list(number),+integer,-list(pair)) - one
```

---

valid_linear_encoders/1

True when a list of encoders only contains valid `continuous/3` encoder terms with distinct attributes.

Compilation flags:
`static`

Template:
`valid_linear_encoders(Encoders)`
Mode and number of proofs:
`valid_linear_encoders(+list(compound))` - `zero_or_one`

---

valid_projection_components/2

True when projection components are numeric vectors compatible with the encoder feature dimension.

Compilation flags:
`static`

Template:
`valid_projection_components(Encoders,Components)`
Mode and number of proofs:
`valid_projection_components(+list(compound),+list(list(number)))` - `zero_or_one`

---

valid_dimension_reducer_metadata/1

True when diagnostics metadata records the reducer model and effective training options.

Compilation flags:
`static`

Template:
`valid_dimension_reducer_metadata(Diagnostics)`
Mode and number of proofs:
`valid_dimension_reducer_metadata(+list(compound))` - `zero_or_one`

---

print_dimension_reducer_details/3

Prints the common diagnostics, encoders, and component-count lines used by reducer-specific pretty printers.

Compilation flags:
`static`

Template:
`print_dimension_reducer_details(Diagnostics,Encoders,Components)`
Mode and number of proofs:
`print_dimension_reducer_details(+list(compound),+list(compound),+list)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.73.2 dimension_reducer_protocol

Protocol for machine learning dimension reducers.

Availability:
`logtalk_load(dimension_reduction_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-30

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/3*

- learn/2
- transform/3
- check_dimension_reducer/1
- valid_dimension_reducer/1
- diagnostics/2
- diagnostic/2
- dimension_reducer_options/2
- export_to_clauses/4
- export_to_file/4
- print_dimension_reducer/1

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/3

Learns a dimension reducer from the given dataset object using the given options.

Compilation flags:

```
static
```

Template:

```
learn(Dataset,DimensionReducer,Options)
```

Mode and number of proofs:

```
learn(+object_identifier,-compound,++list(compound)) - one
```

---

learn/2

Learns a dimension reducer from the given dataset object using default options.

Compilation flags:

```
static
```

Template:

```
learn(Dataset,DimensionReducer)
```

Mode and number of proofs:

```
learn(+object_identifier,-compound) - one
```

---

transform/3

Transforms a new instance into a reduced representation using the learned dimension reducer. The instance is a list of `Attribute-Value` pairs and the reduced representation is a list of `Component-Value` pairs.

Compilation flags:
`static`

Template:
`transform(DimensionReducer,Instance,ReducedInstance)`
Mode and number of proofs:
`transform(+compound,+list,-list(pair))` - `one`

---

check_dimension_reducer/1

Checks that a learned dimension reducer term is structurally valid for the receiving implementation. Throws an exception when the term is not a valid dimension reducer representation.

Compilation flags:
`static`

Template:
`check_dimension_reducer(DimensionReducer)`
Mode and number of proofs:
`check_dimension_reducer(@compound)` - `one_or_error`

Exceptions:
`DimensionReducer` is a variable:
`instantiation_error`
`DimensionReducer` is neither a variable nor a valid dimension reducer:
`domain_error(dimension_reducer,DimensionReducer)`

---

valid_dimension_reducer/1

True when a learned dimension reducer term is structurally valid for the receiving implementation. Succeeds iff `check_dimension_reducer/1` succeeds without throwing an exception.

Compilation flags:
`static`

Template:
`valid_dimension_reducer(DimensionReducer)`
Mode and number of proofs:
`valid_dimension_reducer(@compound)` - `zero_or_one`

---

diagnostics/2

Returns diagnostics and metadata associated with a learned dimension reducer in a representation-independent way.

Compilation flags:
```
static
```

Template:
```
diagnostics(DimensionReducer,Diagnostics)
```
Mode and number of proofs:
```
diagnostics(+compound,-list(compound)) - one
```

---

diagnostic/2

Tests or enumerates individual diagnostics metadata terms for a learned dimension reducer.

Compilation flags:
```
static
```

Template:
```
diagnostic(DimensionReducer,Diagnostic)
```
Mode and number of proofs:
```
diagnostic(+compound,?compound) - zero_or_more
```

---

dimension_reducer_options/2

Returns the effective training options recorded in a learned dimension reducer diagnostics list.

Compilation flags:
```
static
```

Template:
```
dimension_reducer_options(DimensionReducer,Options)
```
Mode and number of proofs:
```
dimension_reducer_options(+compound,-list(compound)) - zero_or_one
```

---

export_to_clauses/4

Converts a dimension reducer into a list of predicate clauses. `Functor` is the functor for the generated single-argument predicate clauses whose argument is the serialized dimension reducer term.

Compilation flags:

```
static
```

Template:

```
export_to_clauses(Dataset,DimensionReducer,Functor,Clauses)
```

Mode and number of proofs:

```
export_to_clauses(+object_identifier,+compound,+callable,-list(clause)) - one
```

---

export_to_file/4

Exports a dimension reducer to a file. `Functor` is the functor for the generated single-argument predicate clauses whose argument is the serialized dimension reducer term.

Compilation flags:

```
static
```

Template:

```
export_to_file(Dataset,DimensionReducer,Functor,File)
```

Mode and number of proofs:

```
export_to_file(+object_identifier,+compound,+callable,+atom) - one
```

---

print_dimension_reducer/1

Prints a dimension reducer to the current output stream in a human-readable format.

Compilation flags:

```
static
```

Template:

```
print_dimension_reducer(DimensionReducer)
```

Mode and number of proofs:

```
print_dimension_reducer(+compound) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

dimension_reduction_dataset_protocol, *supervised_dimension_reduction_dataset_protocol*, *target_supervised_dimension_reduction_dataset_protocol*

protocol

1.73.3 `dimension_reduction_dataset_protocol`

Protocol for datasets used with unsupervised dimension reduction algorithms.

Availability:
`logtalk_load(dimension_reduction_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-20

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - `attribute_values/2`
  - `example/2`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

attribute_values/2

Enumerates by backtracking the attributes and their possible values. For discrete attributes, `Values` is a list of possible values. For continuous (numeric) attributes, `Values` is the atom `continuous`.

Compilation flags:

```
static
```

Template:

```
attribute_values(Attribute,Values)
```

Mode and number of proofs:

```
attribute_values(?atom,-list(atom)) - zero_or_more
attribute_values(?atom,-atom) - zero_or_more
```

---

example/2

Enumerates by backtracking the examples in the dataset. Each example has an `Id` and a list of `Attribute-Value` pairs.

Compilation flags:

```
static
```

Template:

```
example(Id,AttributeValues)
```

Mode and number of proofs:

```
example(-integer,-list(pair)) - zero_or_more
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

protocol

1.73.4 supervised_dimension_reduction_dataset_protocol

Protocol for labeled datasets used with supervised dimension reduction algorithms such as LDA.

Availability:
`logtalk_load(dimension_reduction_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-18

Compilation flags:
`static`

Extends:
`public` *dimension_reduction_dataset_protocol*

Remarks:
(none)

Inherited public predicates:
attribute_values/2 *example/2*

- *Public predicates*
  - *class/1*
  - *class_values/1*
  - *example/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

class/1

Returns the name of the target class attribute.

Compilation flags:
`static`

Template:
`class(Class)`
Mode and number of proofs:
`class(-atom)` - `one`

class_values/1

Returns the list of possible values for the target class attribute.

Compilation flags:
`static`

Template:
`class_values(Values)`
Mode and number of proofs:
`class_values(-list(atom))` - `one`

---

example/3

Enumerates by backtracking the labeled examples in the dataset. Each example has an `Id`, a `Class` value, and a list of `Attribute-Value` pairs.

Compilation flags:
`static`

Template:
`example(Id,Class,AttributeValues)`
Mode and number of proofs:
`example(-integer,-atom,-list(pair))` - `zero_or_more`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.73.5 target_supervised_dimension_reduction_dataset_protocol

Protocol for target-valued datasets used with supervised dimension reduction algorithms such as PLS projections.

Availability:
`logtalk_load(dimension_reduction_protocols(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-04-28

Compilation flags:
`static`

Extends:
`public` *dimension_reduction_dataset_protocol*

Remarks:
(none)

Inherited public predicates:
attribute_values/2 *example/2*

- *Public predicates*
  - *target/1*
  - *example/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

target/1

Returns the name of the numeric target attribute.

Compilation flags:
`static`

Template:
`target(Target)`
Mode and number of proofs:
`target(-atom)` - `one`

---

example/3

Enumerates by backtracking the target-valued examples in the dataset. Each example has an `Id`, a numeric `Target` value, and a list of `Attribute-Value` pairs.

Compilation flags:
`static`

Template:

```
example(Id,Target,AttributeValues)
```

Mode and number of proofs:

```
example(-integer,-number,-list(pair)) - zero_or_more
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.74 doclet

object

1.74.1 `doclet`

Utility object to help automate (re)generating documentation for a project.

Availability:

```
logtalk_load(doclet(loader))
```

Author: Paulo Moura
Version: 0:5:0
Date: 2017-01-05

Compilation flags:

```
static, context_switching_calls
```

Provides:

logtalk::message_tokens//2

Uses:

logtalk
os

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *update/0*
  - *doc_goal/1*
  - *shell_command/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

update/0

Updates the project documentation, first by calling a sequence of goals and second by executing a sequence of shell commands. Fails if any goal or shell command fails.

Compilation flags:
`static`

Mode and number of proofs:
`update` - `zero_or_one`

---

doc_goal/1

Table of goals, typically using the `diagrams` and the `lgtdoc` tools, used to generate the documentation. Goals are called in the order they are defined and in the context of the `user` pseudo-object.

Compilation flags:
`static`

Template:
`doc_goal(Goal)`
Mode and number of proofs:
`doc_goal(?callable)` - `one_or_more`

---

shell_command/1

Table of shell commands to convert intermediate documentation files into user-friendly documentation. Commands are executed in the order they are defined.

Compilation flags:
`static`

Template:
`shell_command(Command)`

Mode and number of proofs:
`shell_command(?atom)` - `one_or_more`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

lgtdocp, *diagram(Format)*

1.75 eclat_pattern_miner

object

1.75.1 `eclat_pattern_miner`

Eclat frequent itemset miner for transaction datasets using portable vertical tidsets and depth-first recursive prefix extension.

Availability:
`logtalk_load(eclat_pattern_miner(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *frequent_pattern_mining_common*
Uses:
avltree
format
list

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_pattern_miner/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *mine/2* *mine/3* *option/2* *option/3* *pattern_miner_options/2* *print_pattern_miner/1* *valid_option/1* *valid_options/1* *valid_pattern_miner/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pattern_miner_protocol, *transaction_dataset_protocol*, *apriori_pattern_miner*, *fp_growth_pattern_miner*

1.76 edcg

object

1.76.1 `edcg`

Multiple hidden parameters: an extension to Prolog's DCG notation. Ported to Logtalk as a hook object.

Availability:

```
logtalk_load(edcg(loader))
```

Author: Peter Van Roy; adapted to Logtalk by Paulo Moura.
Version: 1:4:2
Date: 2020-04-08

Copyright: Copyright (C) 1992 Peter Van Roy
License: MIT

Compilation flags:

`static, context_switching_calls`

Implements:
`public` *expanding*
Provides:
logtalk::message_tokens//2
Uses:
list
logtalk

Remarks:

- Usage: Compile source files with objects (or categories) defining EDCGs using the compiler option `hook(edcg)`.

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *pred_info/3*
  - *acc_info/7*
  - *acc_info/5*
  - *pass_info/2*
  - *pass_info/1*
- *Operators*
  - *op(1200,xfx,-->>)*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

pred_info/3

Declares predicates that have the listed hidden parameters.

Compilation flags:
`dynamic`

Template:
`pred_info(Name,Arity,HiddenParameters)`
Mode and number of proofs:

`pred_info(?atom,?integer,?list(atom))` - `zero_or_more`

---

acc_info/7

Long form for declaring accumulators.

Compilation flags:
`dynamic`

Template:
`acc_info(Accumulator,Term,Left,Right,Joiner,LStart,RStart)`
Mode and number of proofs:
`acc_info(?atom,?term,?term,?term,?callable,?term,?term)` - `zero_or_more`

---

acc_info/5

Short form for declaring accumulators.

Compilation flags:
`dynamic`

Template:
`acc_info(Accumulator,Term,Left,Right,Joiner)`
Mode and number of proofs:
`acc_info(?atom,?term,?term,?term,?callable)` - `zero_or_more`

---

pass_info/2

Long form for declaring passed arguments. Passed arguments are conceptually the same as accumulators with `=/2` as the joiner function.

Compilation flags:
`dynamic`

Template:
`pass_info(Argument,PStart)`
Mode and number of proofs:
`pass_info(?atom,?term)` - `zero_or_more`

---

pass_info/1

Short form for declaring passed arguments. Passed arguments are conceptually the same as accumulators with =/2 as the joiner function.

Compilation flags:
dynamic

Template:
pass_info(Argument)
Mode and number of proofs:
pass_info(?atom) - zero_or_more

Operators

op(1200,xfx,-->>)

Scope:
public

1.77 elastic_net_regression

object

1.77.1 elastic_net_regression

Elastic net regression regressor supporting continuous and mixed-feature datasets using coordinate descent with coefficient-wise L1 shrinkage plus L2 stabilization. Learns from a dataset object implementing the regression_dataset_protocol protocol and returns a regressor term that can be used for prediction and exported as predicate clauses.

Availability:
logtalk_load(elastic_net_regression(loader))

Author: Paulo Moura
Version: 1:1:1
Date: 2026-10-13

Compilation flags:
static, context_switching_calls

Imports:
public *regressor_common*
Uses:
format
list
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

linear_regression, *ridge_regression*, *lasso_regression*, *knn_regression*, *regression_tree*, *random_forest_regression*, *gradient_boosting_regression*

1.78 elo_ranker

object

1.78.1 `elo_ranker`

Elo pairwise preference ranker. Learns one deterministic rating per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol by replaying the observed preference stream using a deterministic batch Elo update rule, and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:
`logtalk_load(elo_ranker(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-05-06

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *ranking_dataset_common*

`public` *score_ranker_model_common*

Uses:

avltree

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *colley_ranker*, *rank_centrality*

1.79 emeans_clusterer

object

1.79.1 `emeans_clusterer`

Convenience object for `emeans_clusterer/1` using the `fast_random(xoshiro128pp)` random number generator.

Availability:
`logtalk_load(emeans_clusterer(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *emeans_clusterer(fast_random(xoshiro128pp))*

Remarks:
(none)

Inherited public predicates:
check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

emeans_clusterer(Random)

object

1.79.2 `emeans_clusterer(Random)`

- `Random` - Random number generator object providing `random/1`, `randomize/1`, `get_seed/1`, and `set_seed/1` predicates.

e-Means clusterer for continuous datasets. Defaults to the published EMA training algorithm with count-weighted donor splitting and last-epoch-mean finalization. Also supports slow-EMA finalization and a hybrid of EMA training and bounded Lloyd refinement. Learns from a dataset object implementing the `clustering_dataset_protocol` protocol.

Availability:
`logtalk_load(emeans_clusterer(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *clusterer_common*
Uses:
avltree
format

list
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *nearest_centroid_classifier*, *kmeans_clusterer*

1.80 events

object

1.80.1 after_event_registry

After events registry predicates.

Availability:
`logtalk_load(events(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2009-10-08

Compilation flags:
`static, context_switching_calls, events`

Implements:
`public` *event_registryp*

Remarks:
(none)

Inherited public predicates:
del_monitors/0 *del_monitors/4* *monitor/1* *monitor/4* *monitored/1* *monitors/1* *set_monitor/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

before_event_registry, *monitorp*

object

1.80.2 before_event_registry

Before events registry predicates.

Availability:
`logtalk_load(events(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2009-10-08

Compilation flags:
`static, context_switching_calls, events`

Implements:
`public` *event_registryp*

Remarks:
(none)

Inherited public predicates:
del_monitors/0 *del_monitors/4* *monitor/1* *monitor/4* *monitored/1* *monitors/1* *set_monitor/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

after_event_registry, *monitorp*

object

1.80.3 event_registry

Before and after events registry predicates.

Availability:
`logtalk_load(events(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2009-10-08

Compilation flags:
`static, context_switching_calls, events`

Implements:
`public` *event_registryp*

Remarks:
(none)

Inherited public predicates:
del_monitors/0 *del_monitors/4* *monitor/1* *monitor/4* *monitored/1* *monitors/1* *set_monitor/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.80.4 event_registryp

Event registry protocol.

Availability:
`logtalk_load(events(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2009-10-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *monitors/1*
  - *monitor/1*
  - *monitored/1*
  - *monitor/4*
  - *set_monitor/4*
  - *del_monitors/4*
  - *del_monitors/0*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

monitors/1

Returns a list of all current monitors.

Compilation flags:
`static`

Template:
`monitors(Monitors)`
Mode and number of proofs:
`monitors(-list(object_identifier))-one`

monitor/1

Monitor is an object playing the role of a monitor.

Compilation flags:
```
static
```

Template:
```
monitor(Monitor)
```
Mode and number of proofs:
```
monitor(-object_identifier) - zero_or_more
monitor(+object_identifier) - zero_or_one
```

---

monitored/1

Returns a list of all currently monitored objects.

Compilation flags:
```
static
```

Template:
```
monitored(Objects)
```
Mode and number of proofs:
```
monitored(-list(object_identifier)) - one
```

---

monitor/4

True if the arguments describe a currently defined monitored event.

Compilation flags:
```
static
```

Template:
```
monitor(Object,Message,Sender,Monitor)
```
Mode and number of proofs:
```
monitor(?object_identifier,?nonvar,?object_identifier,?object_identifier) -
zero_or_more
```

---

set_monitor/4

Sets a monitor for the set of matching events.

Compilation flags:
`static`

Template:
`set_monitor(Object,Message,Sender,Monitor)`
Mode and number of proofs:
`set_monitor(?object_identifier,?nonvar,?object_identifier,+object_identifier)` - `zero_or_one`

---

del_monitors/4

Deletes all matching monitored events.

Compilation flags:
`static`

Template:
`del_monitors(Object,Message,Sender,Monitor)`
Mode and number of proofs:
`del_monitors(?object_identifier,?nonvar,?object_identifier,?object_identifier)` - `one`

---

del_monitors/0

Deletes all monitored events.

Compilation flags:
`static`

Mode and number of proofs:
`del_monitors` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

event_registry, *monitorp*

category

1.80.5 `monitor`

Monitor predicates.

Availability:
`logtalk_load(events(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2019-03-08

Compilation flags:
`static, events`

Implements:
`public` *monitorp*

Remarks:
(none)

Inherited public predicates:
activate_monitor/0 *del_spy_points/4* *monitor_activated/0* *reset_monitor/0* *set_spy_point/4* *spy_point/4* *suspend_monitor/0*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *spy_point_/4*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

spy_point_/4

Stores current spy points.

Compilation flags:
`dynamic`

Template:
`spy_point_(Event,Object,Message,Sender)`
Mode and number of proofs:
`spy_point_(?event,?object,?callable,?object)` - `zero_or_more`

Operators

(none)
protocol

1.80.6 monitorp

Monitor protocol.

Availability:
`logtalk_load(events(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2000-07-24

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *monitor_activated/0*
  - *activate_monitor/0*
  - *suspend_monitor/0*
  - *reset_monitor/0*
  - *spy_point/4*
  - *set_spy_point/4*
  - *del_spy_points/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

monitor_activated/0

True if monitor is currently active.

Compilation flags:

```
static
```

Mode and number of proofs:

```
monitor_activated - zero_or_one
```

---

activate_monitor/0

Activates all spy points and start monitoring.

Compilation flags:

```
static
```

Mode and number of proofs:

```
activate_monitor - one
```

---

suspend_monitor/0

Suspends monitoring, deactivating all spy points.

Compilation flags:

```
static
```

Mode and number of proofs:

```
suspend_monitor - one
```

---

reset_monitor/0

Resets monitor, deactivating and deleting all spy points.

Compilation flags:
`static`

Mode and number of proofs:
`reset_monitor` - `one`

---

spy_point/4

Current spy point.

Compilation flags:
`static`

Template:
`spy_point(Event,Object,Message,Sender)`
Mode and number of proofs:
`spy_point(?event,?object,?callable,?object)` - `zero_or_more`

---

set_spy_point/4

Sets a spy point.

Compilation flags:
`static`

Template:
`set_spy_point(Event,Object,Message,Sender)`
Mode and number of proofs:
`set_spy_point(?event,?object,?callable,?object)` - `one`

---

del_spy_points/4

Deletes all matching spy points.

Compilation flags:
`static`

Template:

```
del_spy_points(Event,Object,Message,Sender)
```

Mode and number of proofs:

```
del_spy_points(@event,@object,@callable,@object) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

monitor, *event_registryp*

1.81 ewma_anomaly_detector

object

1.81.1 `ewma_anomaly_detector`

EWMA (Exponentially Weighted Moving Average) anomaly detector for continuous sequence-like datasets. Learns per-step population mean and standard deviation from baseline training examples selected from a dataset object implementing the `anomaly_dataset_protocol` protocol and returns a detector term that can be used for scoring, prediction, and export.

Availability:

```
logtalk_load(ewma_anomaly_detector(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public anomaly_detector_common
```

Uses:

format
list
pairs
population
type

Remarks:
(none)

Inherited public predicates:
anomaly_detector_options/2 *check_anomaly_detector/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict/4* *print_anomaly_detector/1* *score/3* *score_all/3* *valid_anomaly_detector/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

anomaly_dataset_protocol, *anomaly_detector_protocol*, *cusum_anomaly_detector*, *isolation_forest_anomaly_detector*, *knn_distance_anomaly_detector*, *lof_anomaly_detector*, *modified_z_score_anomaly_detector*, *z_score_anomaly_detector*

1.82 expand_library_alias_paths

object

1.82.1 `expand_library_alias_paths`

Hook object for expanding library alias paths in `logtalk_library_path/2` facts when compiling a source file.

Availability:
`logtalk_load(expand_library_alias_paths(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2018-04-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Uses:
logtalk
os

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.83 expecteds

object

1.83.1 `either`

Types and predicates for extended type-checking and handling of expected terms.

Availability:
`logtalk_load(expecteds(loader))`

Author: Paulo Moura
Version: 0:9:0
Date: 2025-06-19

Compilation flags:
`static, context_switching_calls`

Provides:
type::type/1
type::check/2
arbitrary::arbitrary/1
arbitrary::arbitrary/2
arbitrary::shrinker/1
arbitrary::shrink/3
arbitrary::edge_case/2

Uses:
expected
expected(Expected)
random
type

Remarks:

- Type-checking support: Defines a `either(ValueType, ErrorType)` type for checking expected terms where the value and error terms must be of the given types.
- QuickCheck support: Defines clauses for the `type::arbitrary/1-2`, `arbitrary::shrinker/1`, `arbitrary::shrink/3`, and `arbitrary::edge_case/2` predicates to allow generating random values for the `either(ValueType, ErrorType)` type.

Inherited public predicates:
(none)

- *Public predicates*
  - *expecteds/2*
  - *unexpecteds/2*
  - *partition/3*
  - *sequence/2*
  - *traverse/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

expecteds/2

Returns the values stored in the expected terms that hold a value.

Compilation flags:
```
static
```

Template:
```
expecteds(Expecteds,Values)
```
Mode and number of proofs:
```
expecteds(+list(expected),-list) - one
```

---

unexpecteds/2

Returns the errors stored in the expected terms that hold an error.

Compilation flags:
```
static
```

Template:
```
unexpecteds(Expecteds,Errors)
```
Mode and number of proofs:
```
unexpecteds(+list(expected),-list) - one
```

---

partition/3

Retrieves and partitions the values and errors hold by the expected terms.

Compilation flags:
```
static
```

Template:
```
partition(Expecteds,Values,Errors)
```
Mode and number of proofs:
```
partition(+list(expected),-list,-list) - one
```

---

sequence/2

Returns an expected term with a list of all values when all expected terms hold values. Otherwise returns the first expected term holding an error.

Compilation flags:

```
static
```

Template:

```
sequence(Expecteds,Expected)
```

Mode and number of proofs:

```
sequence(+list(expected),--nonvar) - one
```

---

traverse/3

Applies a closure to each list element to generate expected terms and then sequences them into a single expected term holding all values or the first error.

Compilation flags:

```
static
```

Template:

```
traverse(Closure,Terms,Expected)
```

Meta-predicate template:

```
traverse(2,*,*)
```

Mode and number of proofs:

```
traverse(+callable,+list,--nonvar) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

expected, *expected(Expected)*, *type*, *arbitrary*

object

1.83.2 expected

Constructors for expected terms. An expected term contains either a value or an error. Expected terms should be regarded as opaque terms and always used with the `expected/1` object by passing the expected term as a parameter.

Availability:
`logtalk_load(expecteds(loader))`

Author: Paulo Moura
Version: 2:2:0
Date: 2026-02-21

Compilation flags:
`static, context_switching_calls`

Provides:
type::type/1
type::check/2

Remarks:

- Type-checking support: This object also defines a type `expected` for use with the `type` library object.

Inherited public predicates:
(none)

- *Public predicates*
  - *of_unexpected/2*
  - *of_expected/2*
  - *from_goal/4*
  - *from_goal/3*
  - *from_goal/2*
  - *from_generator/4*
  - *from_generator/3*
  - *from_generator/2*
  - *from_optional/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

of_unexpected/2

Constructs an expected term from an error that represent that the expected value is missing.

Compilation flags:
`static`

Template:
`of_unexpected(Error,Expected)`
Mode and number of proofs:
`of_unexpected(@term,--nonvar)` - `one`

---

of_expected/2

Constructs an expected term from an expected value.

Compilation flags:
`static`

Template:
`of_expected(Value,Expected)`
Mode and number of proofs:
`of_expected(@term,--nonvar)` - `one`

---

from_goal/4

Constructs an expected term holding a value bound by calling the given goal. Otherwise returns an expected term with the unexpected goal error or failure represented by the `Error` argument.

Compilation flags:
`static`

Template:
`from_goal(Goal,Value,Error,Expected)`
Meta-predicate template:
`from_goal(0,*,*,*)`
Mode and number of proofs:
`from_goal(+callable,--term,@term,--nonvar)` - `one`

---

from_goal/3

Constructs an expected term holding a value bound by calling the given goal. Otherwise returns an expected term with the unexpected goal error or the atom `fail` representing the unexpected failure.

Compilation flags:
`static`

Template:
`from_goal(Goal,Value,Expected)`
Meta-predicate template:
`from_goal(0,*,*)`
Mode and number of proofs:
`from_goal(+callable,--term,--nonvar)` - `one`

---

from_goal/2

Constructs an expected term holding a value bound by calling the given closure. Otherwise returns an expected term holding the unexpected closure error or the atom `fail` representing the unexpected failure.

Compilation flags:
`static`

Template:
`from_goal(Closure,Expected)`
Meta-predicate template:
`from_goal(1,*)`
Mode and number of proofs:
`from_goal(+callable,--nonvar)` - `one`

---

from_generator/4

Constructs expected terms with the values generated by calling the given goal. On goal error or failure, returns an expected term with the unexpected goal error or failure represented by the `Error` argument.

Compilation flags:
`static`

Template:
`from_generator(Goal,Value,Error,Expected)`
Meta-predicate template:
`from_generator(0,*,*,*)`
Mode and number of proofs:
`from_generator(+callable,--term,@term,--nonvar)` - `one_or_more`

---

from_generator/3

Constructs expected terms with the values generated by calling the given goal. On goal error or failure, returns an expected term with, respectively, the unexpected goal error or the atom `fail` representing the unexpected goal failure.

Compilation flags:
`static`

Template:
`from_generator(Goal,Value,Expected)`
Meta-predicate template:
`from_generator(0,*,*)`
Mode and number of proofs:
`from_generator(+callable,--term,--nonvar)` - `one_or_more`

---

from_generator/2

Constructs expected terms with the values generated by calling the given closure. On closure error or failure, returns an expected term with, respectively, the unexpected closure error or the atom `fail` representing the unexpected closure failure.

Compilation flags:
`static`

Template:
`from_generator(Closure,Expected)`
Meta-predicate template:
`from_generator(1,*)`
Mode and number of proofs:
`from_generator(+callable,--nonvar)` - `one_or_more`

---

from_optional/3

Converts an optional term to an expected term. Returns an expected term holding the value if the optional term is not empty. Returns an expected term with the given error otherwise.

Compilation flags:
`static`

Template:
`from_optional(Optional,Error,Expected)`
Mode and number of proofs:
`from_optional(+nonvar,@term,--nonvar)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

expected(Expected), *type*

object

1.83.3 `expected(Expected)`

Expected term predicates. Requires passing an expected term (constructed using the `expected` object predicates) as a parameter.

Availability:

```
logtalk_load(expecteds(loader))
```

Author: Paulo Moura
Version: 1:6:0
Date: 2026-02-21

Compilation flags:

```
static, context_switching_calls
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *is_expected/0*
  - *is_unexpected/0*
  - *if_expected/1*
  - *if_unexpected/1*
  - *if_expected_or_else/2*
  - *unexpected/1*
  - *expected/1*

      - *map/2*
      - *flat_map/2*
      - *either/3*
      - *or_else/2*
      - *or_else_get/2*
      - *or_else_call/2*
      - *or_else_throw/1*
      - *or_else_fail/1*
      - *filter/3*
      - *map_unexpected/2*
      - *map_catching/2*
      - *map_both/3*
      - *swap/1*
      - *map_or_else/3*
      - *or/2*
      - *or_else_throw/2*
      - *zip/3*
      - *to_optional/1*
      - *flatten/1*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

is_expected/0

True if the expected term holds a value. See also the `if_expected/1` predicate.

Compilation flags:
`static`

Mode and number of proofs:
`is_expected - zero_or_one`

---

is_unexpected/0

True if the expected term holds an error. See also the `if_unexpected/1` predicate.

Compilation flags:
`static`

Mode and number of proofs:
`is_unexpected - zero_or_one`

---

if_expected/1

Applies a closure when the expected term holds a value using the value as argument. Succeeds otherwise.

Compilation flags:
`static`

Template:
`if_expected(Closure)`
Meta-predicate template:
`if_expected(1)`
Mode and number of proofs:
`if_expected(+callable)` - `zero_or_more`

---

if_unexpected/1

Applies a closure when the expected term holds an error using the error as argument. Succeeds otherwise. Can be used to throw the exception hold by the expected term by calling it the atom `throw`.

Compilation flags:
`static`

Template:
`if_unexpected(Closure)`
Meta-predicate template:
`if_unexpected(1)`
Mode and number of proofs:
`if_unexpected(+callable)` - `zero_or_more`

---

if_expected_or_else/2

Applies either `ExpectedClosure` or `UnexpectedClosure` depending on the expected term holding a value or an error.

Compilation flags:
`static`

Template:
`if_expected_or_else(ExpectedClosure,UnexpectedClosure)`
Meta-predicate template:
`if_expected_or_else(1,1)`
Mode and number of proofs:
`if_expected_or_else(+callable,+callable)` - `zero_or_more`

---

unexpected/1

Returns the error hold by the expected term. Throws an error otherwise.

Compilation flags:
`static`

Template:
`unexpected(Error)`
Mode and number of proofs:
`unexpected(--term) - one_or_error`

Exceptions:
Expected term holds a value:
`existence_error(unexpected_error,Expected)`

---

expected/1

Returns the value hold by the expected term. Throws an error otherwise.

Compilation flags:
`static`

Template:
`expected(Value)`
Mode and number of proofs:
`expected(--term) - one_or_error`

Exceptions:
Expected term holds an error:
`existence_error(expected_value,Expected)`

---

map/2

When the expected term does not hold an error and mapping a closure with the expected value and the new value as additional arguments is successful, returns an expected term with the new value. Otherwise returns the same expected term.

Compilation flags:
`static`

Template:
`map(Closure,NewExpected)`
Meta-predicate template:
`map(2,*)`
Mode and number of proofs:

```
map(+callable,--nonvar) - one
```

---

flat_map/2

When the expected term does not hold an error and mapping a closure with the expected value and the new expected term as additional arguments is successful, returns the new expected term. Otherwise returns the same expected term.

Compilation flags:

```
static
```

Template:

```
flat_map(Closure,NewExpected)
```

Meta-predicate template:

```
flat_map(2,*)
```

Mode and number of proofs:

```
flat_map(+callable,--nonvar) - one
```

---

either/3

Applies either `ExpectedClosure` if the expected term holds a value or `UnexpectedClosure` if the expected term holds an error. Returns a new expected term if the applied closure is successful. Otherwise returns the same expected term.

Compilation flags:

```
static
```

Template:

```
either(ExpectedClosure,UnexpectedClosure,NewExpected)
```

Meta-predicate template:

```
either(2,2,*)
```

Mode and number of proofs:

```
either(+callable,+callable,--nonvar) - one
```

---

or_else/2

Returns the value hold by the expected term if it does not hold an error or the given default term if the expected term holds an error.

Compilation flags:

```
static
```

Template:

```
or_else(Value,Default)
```

Mode and number of proofs:

```
or_else(--term,@term) - one
```

---

or_else_get/2

Returns the value hold by the expected term if it does not hold an error. Otherwise applies a closure to compute the expected value. Throws an error when the expected term holds an error and a value cannot be computed.

Compilation flags:
`static`

Template:
`or_else_get(Value,Closure)`
Meta-predicate template:
`or_else_get(*,1)`
Mode and number of proofs:
`or_else_get(--term,+callable)` - `one_or_error`

Exceptions:
Expected term holds an unexpected error and an expected value cannot be computed:
`existence_error(expected_value,Expected)`

---

or_else_call/2

Returns the value hold by the expected term if it does not hold an error. Calls a goal deterministically otherwise.

Compilation flags:
`static`

Template:
`or_else_call(Value,Goal)`
Meta-predicate template:
`or_else_call(*,0)`
Mode and number of proofs:
`or_else_call(--term,+callable)` - `zero_or_one`

---

or_else_throw/1

Returns the value hold by the expected term if present. Throws the error hold by the expected term as an exception otherwise.

Compilation flags:
`static`

Template:
`or_else_throw(Value)`
Mode and number of proofs:
`or_else_throw(--term)` - `one_or_error`

---

or_else_fail/1

Returns the value hold by the expected term if it does not hold an error. Fails otherwise. Usually called to skip over expected terms holding errors.

Compilation flags:

```
static
```

Template:

```
or_else_fail(Value)
```

Mode and number of proofs:

```
or_else_fail(--term) - zero_or_one
```

---

filter/3

When the expected term holds a value and the value satisfies the closure, returns the same expected term. When the expected term holds a value that does not satisfy the closure, returns an expected term with the given error. When the expected term holds an error, returns the same expected term.

Compilation flags:

```
static
```

Template:

```
filter(Closure,Error,NewExpected)
```

Meta-predicate template:

```
filter(1,*,*)
```

Mode and number of proofs:

```
filter(+callable,@term,--nonvar) - one
```

---

map_unexpected/2

When the expected term holds an error and mapping a closure with the error and the new error as additional arguments is successful, returns an expected term with the new error. Otherwise returns the same expected term.

Compilation flags:

```
static
```

Template:

```
map_unexpected(Closure,NewExpected)
```

Meta-predicate template:

```
map_unexpected(2,*)
```

Mode and number of proofs:

```
map_unexpected(+callable,--nonvar) - one
```

---

map_catching/2

When the expected term holds a value, applies a closure to it. Returns an expected term with the new value if the closure succeeds. Returns an expected term with the error if the closure throws an error. Returns an expected term with the atom `fail` as error if the closure fails. When the expected term holds an error, returns the same expected term.

Compilation flags:

```
static
```

Template:

```
map_catching(Closure,NewExpected)
```

Meta-predicate template:

```
map_catching(2,*)
```

Mode and number of proofs:

```
map_catching(+callable,--nonvar) - one
```

---

map_both/3

When the expected term holds a value and mapping `ExpectedClosure` with the value is successful, returns an expected term with the new value. When the expected term holds an error and mapping `UnexpectedClosure` with the error is successful, returns an expected term with the new error. Otherwise returns the same expected term.

Compilation flags:

```
static
```

Template:

```
map_both(ExpectedClosure,UnexpectedClosure,NewExpected)
```

Meta-predicate template:

```
map_both(2,2,*)
```

Mode and number of proofs:

```
map_both(+callable,+callable,--nonvar) - one
```

---

swap/1

Swaps the expected and unexpected terms. If the expected term holds a value, returns an unexpected term with that value. If the expected term holds an error, returns an expected term with that error.

Compilation flags:

```
static
```

Template:

```
swap(NewExpected)
```

Mode and number of proofs:

```
swap(--nonvar) - one
```

---

map_or_else/3

When the expected term holds a value and mapping a closure with the value and the new value as additional arguments is successful, returns the new value. Otherwise returns the given default value.

Compilation flags:
```
static
```

Template:
```
map_or_else(Closure,Default,Value)
```
Meta-predicate template:
```
map_or_else(2,*,*)
```
Mode and number of proofs:
```
map_or_else(+callable,@term,--term) - one
```

---

or/2

Returns the same expected term if it holds a value. Otherwise calls closure to generate a new expected term. Fails if the expected term holds an error and calling the closure fails or throws an error.

Compilation flags:
```
static
```

Template:
```
or(NewExpected,Closure)
```
Meta-predicate template:
```
or(*,1)
```
Mode and number of proofs:
```
or(--term,@callable) - zero_or_one
```

---

or_else_throw/2

Returns the value hold by the expected term if present. Throws the given error otherwise, ignoring any error hold by the expected term.

Compilation flags:
```
static
```

Template:
```
or_else_throw(Value,Error)
```
Mode and number of proofs:
```
or_else_throw(--term,@nonvar) - one_or_error
```

---

zip/3

When both this expected and the other expected hold values and applying a closure with both values and the new value as additional arguments is successful, returns an expected term with the new value. Otherwise returns the first expected term that holds an error.

Compilation flags:
`static`

Template:
`zip(Closure,OtherExpected,NewExpected)`
Meta-predicate template:
`zip(3,*,*)`
Mode and number of proofs:
`zip(+callable,+nonvar,--nonvar)` - `one`

---

to_optional/1

Converts the expected term to an optional term. Returns a non-empty optional term holding the value if the expected term holds a value. Returns an empty optional term if the expected term holds an error.

Compilation flags:
`static`

Template:
`to_optional(Optional)`
Mode and number of proofs:
`to_optional(--nonvar)` - `one`

---

flatten/1

Flattens a nested expected term. When the expected term holds a value that is itself an expected term, returns the inner expected term. When the expected term holds a non-expected value, returns the same expected term. When the expected term holds an error, returns the same expected term.

Compilation flags:
`static`

Template:
`flatten(NewExpected)`
Mode and number of proofs:
`flatten(--nonvar)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

expected

1.84 exponential_smoothing

object

1.84.1 `exponential_smoothing`

Simple, Holt linear-trend, and additive or multiplicative Holt-Winters exponential smoothing forecaster.

Availability:
`logtalk_load(exponential_smoothing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-24

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *forecaster_common*
`public` *exponential_smoothing_common*

Uses:
differential_evolution(Problem)
fast_random(Algorithm)
format
integer
list
nelder_mead(Problem)
numberlist
type

Remarks:
(none)

Inherited public predicates:

check_forecaster/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *forecast/3* *forecaster_options/2* *learn/2* *learn/3* *option/2* *option/3* *print_forecaster/1* *valid_forecaster/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *update/3*
  - *forecast_interval/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

update/3

Returns a new forecaster after applying one numeric observation or, under skip_update, an unbound variable representing a missing observation. Does not bind missing observations.

Compilation flags:

`static`

Template:

`update(Forecaster,Observation,UpdatedForecaster)`

Mode and number of proofs:

`update(+compound,@term,-compound)` - `one_or_error`

Exceptions:

`Forecaster` is a variable:

`instantiation_error`

`Forecaster` is neither a variable nor a valid forecaster:

`domain_error(forecaster,Forecaster)`

`Observation` is a variable under the error missing policy:

`instantiation_error`

`Observation` is neither a variable nor a number:

`type_error(number,Observation)`

`Observation` is a number but is not finite:

`domain_error(finite_number,Observation)`

`Observation` is incompatible with the learned transformation:

`domain_error(positive_transformation_series,Observation)`

`Observation` is incompatible with a multiplicative model:

`domain_error(positive_multiplicative_series,Observation)`

The multiplicative level update is not positive:

`domain_error(positive_multiplicative_level,Level)`

Update or diagnostic arithmetic fails:

`evaluation_error(Error)`

forecast_interval/5

Computes residual-bootstrap prediction interval bounds for the next `Horizon` forecasts of a learned forecaster. Requires a forecaster learned with the `retain_residuals(true)` learning option. For each of `samples/1` simulated paths and each horizon step, independently resamples, with replacement, one retained one-step training residual, adds it to the deterministic point forecast for that step (thus preserving the seasonal phase of the point forecast), and inverse-transforms the simulated value using the same transformation and bias adjustment as `forecast/3`. The reported bounds are the empirical `(1-confidence)/2` and `(1+confidence)/2` quantiles of the simulated values for each step, widened when necessary so that the point forecast always lies within its bounds. A zero horizon returns two empty lists and does not require retained residuals.

Compilation flags:

`static`

Template:

`forecast_interval(Forecaster,Horizon,Lower,Upper,Options)`

Mode and number of proofs:

`forecast_interval(+compound,+non_negative_integer,-list(number),-list(number),+list(compound))` - `one_or_error`

Exceptions:

`Forecaster` is a variable:

`instantiation_error`

`Forecaster` is neither a variable nor a valid forecaster:

`domain_error(forecaster,Forecaster)`

`Horizon` is a variable:

`instantiation_error`

`Horizon` is neither a variable nor an integer:

`type_error(integer,Horizon)`

`Horizon` is an integer but is negative:

`domain_error(non_negative_integer,Horizon)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An option is a variable:

`instantiation_error`

An option is neither a variable nor a compound term:

`type_error(compound,Option)`

An option is a compound term but is not a valid interval option:

`domain_error(option,Option)`

`Horizon` is positive and the forecaster was learned without `retain_residuals(true)`:

`domain_error(retained_residuals,Forecaster)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

forecaster_protocol, *time_series_dataset_protocol*

category

1.84.2 exponential_smoothing_common

Shared fitting and forecasting predicates for exponential smoothing models.

Availability:
`logtalk_load(exponential_smoothing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-02

Compilation flags:
`static`

Uses:
deque
list
numberlist

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *finite_number/1*
  - *observed_series/2*
  - *fit_smoothing/9*
  - *update_smoothing/7*

      - *forecast_smoothing/4*
      - *initial_parameter_point/2*
      - *parameter_bounds/2*
      - *parameters_from_point/3*
      - *optimization_initial_point/6*
      - *optimization_bounds/6*
      - *optimization_components/8*
  - *Private predicates*
  - *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

finite_number/1

True when the argument is a finite integer or floating point number.

Compilation flags:
`static`

Template:
`finite_number(Number)`
Mode and number of proofs:
`finite_number(+term)` - `zero_or_one`

---

observed_series/2

Returns the known observations from a validated numeric-or-unbound series without binding missing observations.

Compilation flags:
`static`

Template:
`observed_series(Series,ObservedSeries)`
Mode and number of proofs:
`observed_series(+list,-list(number))` - `one`

---

fit_smoothing/9

Fits the selected exponential smoothing method using fixed parameters and the requested initialization strategy. Unbound observations advance predictions and seasonal phase without being scored, instantiated, or updating smoothing components.

Compilation flags:

```
static
```

Template:

```
fit_smoothing(Method,Series,Frequency,Parameters,InitializationSpecification,
State,SumSquaredError,ErrorCount,Residuals)
```

Mode and number of proofs:

```
fit_smoothing(+atom,+list,+term,+list(float),+compound,-compound,-float,
-non_negative_integer,-list(float)) - one_or_error
```

Exceptions:

Initialization has too few known observations:

```
domain_error(insufficient_known_observations,Initialization)
```

A seasonal phase has no known initialization observation:

```
domain_error(insufficient_seasonal_phase_observations,Phase)
```

The multiplicative Holt-Winters level update is not positive:

```
domain_error(positive_multiplicative_level,Level)
```

Fitting or initialization arithmetic fails:

```
evaluation_error(Error)
```

---

update_smoothing/7

Updates a learned smoothing state with one observation by applying the same fixed-parameter transition used during batch fitting. The returned residual list is empty for a skipped missing observation and a singleton for an observed value.

Compilation flags:

```
static
```

Template:

```
update_smoothing(Method,State,Parameters,Observation,UpdatedState,
SumSquaredErrorIncrement,Residuals)
```

Mode and number of proofs:

```
update_smoothing(+atom,+compound,+list(float),+term,-compound,-float,
-list(float)) - one_or_error
```

Exceptions:

The multiplicative Holt-Winters level update is not positive:

```
domain_error(positive_multiplicative_level,Level)
```

Update arithmetic fails:

```
evaluation_error(Error)
```

---

forecast_smoothing/4

Generates forecasts for the selected exponential smoothing method from a learned final state.

Compilation flags:
`static`

Template:
`forecast_smoothing(Method,State,Horizon,Forecasts)`
Mode and number of proofs:
`forecast_smoothing(+atom,+compound,+non_negative_integer,-list(number))` - `one_or_error`

Exceptions:
A multiplicative Holt-Winters forecast is not positive:
`domain_error(positive_multiplicative_forecast,Forecast)`

---

initial_parameter_point/2

Builds the optimizer initial point from the parameters whose option value is `auto`.

Compilation flags:
`static`

Template:
`initial_parameter_point(ParameterSpecification,Point)`
Mode and number of proofs:
`initial_parameter_point(+list(compound),-list(float))` - `one`

---

parameter_bounds/2

Builds closed unit-interval bounds for the parameters whose option value is `auto`.

Compilation flags:
`static`

Template:
`parameter_bounds(ParameterSpecification,Bounds)`
Mode and number of proofs:
`parameter_bounds(+list(compound),-list(pair(float)))` - `one`

---

parameters_from_point/3

Combines fixed parameter option values and an optimizer point into the canonical numeric parameter list.

Compilation flags:
```
static
```

Template:
```
parameters_from_point(ParameterSpecification,Point,Parameters)
```
Mode and number of proofs:
```
parameters_from_point(+list(compound),+list(float),-list(float)) - one
```

---

optimization_initial_point/6

Builds the joint optimizer initial point for automatic smoothing parameters and, when requested, the initial model state.

Compilation flags:
```
static
```

Template:
```
optimization_initial_point(Method,Series,Frequency,ParameterSpecification,
InitializationSpecification,Point)
```
Mode and number of proofs:
```
optimization_initial_point(+atom,+list(number),+term,+list(compound),+compound,
-list(float)) - one
```

---

optimization_bounds/6

Builds finite data-derived bounds for the joint smoothing-parameter and initial-state optimization point.

Compilation flags:
```
static
```

Template:
```
optimization_bounds(Method,Series,Frequency,ParameterSpecification,
InitializationSpecification,Bounds)
```
Mode and number of proofs:
```
optimization_bounds(+atom,+list(number),+term,+list(compound),+compound,
-list(pair(float))) - one
```

---

optimization_components/8

Decodes a joint optimization point into the canonical smoothing parameter list and effective initialization specification.

Compilation flags:

```
static
```

Template:

```
optimization_components(Method,Series,Frequency,ParameterSpecification,
InitializationSpecification,Point,Parameters,
EffectiveInitializationSpecification)
```

Mode and number of proofs:

```
optimization_components(+atom,+list(number),+term,+list(compound),+compound,
+list(float),-list(float),-compound) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.84.3 exponential_smoothing_problem(Method,Series,OptimizationSeries,Frequency,ParameterSpecification,InitializationSpecification)

- `Method` - Exponential smoothing method.
- `Series` - Validated training series.
- `OptimizationSeries` - Training series with missing observations removed for optimizer initialization and bounds.
- `Frequency` - Seasonal frequency or `none`.
- `ParameterSpecification` - Canonical list of fixed or automatic parameter options.
- `InitializationSpecification` - Initialization strategy and number of initial cycles.

Bounded sum-of-squared-errors optimization problem for exponential smoothing parameters. Implements both the local-optimization (Nelder-Mead) and Differential Evolution problem protocols over the same objective.

Availability:

```
logtalk_load(exponential_smoothing(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-02

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public local_optimization_problem_protocol
public differential_evolution_problem_protocol
```

Imports:
`public` *exponential_smoothing_common*

Remarks:
(none)

Inherited public predicates:
fitness/2 *gradient/2* *hessian/2* *initial_point/1* *initial_positions/1* *objective/2* *position_bounds/1* *progress/5* *stop_condition/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.85 fcube

object

1.85.1 `fcube`

FCube: An Efficient Prover for Intuitionistic Propositional Logic.

Availability:
`logtalk_load(fcube(loader))`

Author: Mauro Ferrari, Camillo Fiorentini, Guido Fiorino; ported to Logtalk by Paulo Moura.
Version: 5:0:1
Date: 2024-03-14

Copyright: Copyright 2012 Mauro Ferrari, Camillo Fiorentini, Guido Fiorino; Copyright 2020-2024 Paulo Moura
License: GPL-2.0-or-later

Compilation flags:

`static, context_switching_calls`

Uses:

integer
list
os
set
user

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *gnu/0*
  - *fcube/0*
  - *decide/1*
  - *decide/2*
- *Protected predicates*
- *Private predicates*
- *Operators*
  - *op(1200,xfy,<=>)*
  - *op(1110,xfy,=>)*
  - *op(1000,xfy,&&)*
  - *op(500,fy,~)*
  - *op(1100,xfy,v)*

Public predicates

gnu/0

Prints banner with copyright and license information.

Compilation flags:

`static`

Mode and number of proofs:

`gnu - one`

fcube/0

Reads a formula and applies the prover to it, printing its counter-model.

Compilation flags:
`static`

Mode and number of proofs:
`fcube` - `one`

---

decide/1

Applies the prover to the given formula and prints its counter-model.

Compilation flags:
`static`

Template:
`decide(Formula)`
Mode and number of proofs:
`decide(++compound)` - `one`

---

decide/2

Applies the prover to the given formula and returns its counter-model.

Compilation flags:
`static`

Template:
`decide(Formula,CounterModel)`
Mode and number of proofs:
`decide(++compound,--compound)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

op(1200,xfy,<=>)

Scope:
public

op(1110,xfy,=>)

Scope:
public

op(1000,xfy,&&)

Scope:
public

op(500,fy,~)

Scope:
public

op(1100,xfy,v)

Scope:
public

1.86 feature_selection_protocols

object

1.86.1 anova_f_score

Scores a numeric feature against a categorical target using the one-way ANOVA F-statistic over complete examples, capped at 1.0e10. Returns 0.0 for fewer than two classes, no within-class degrees of freedom, or a constant feature. Perfect class separation scores 1.0e10 and ties with any finite statistic reaching the cap.

Availability:
```
logtalk_load(feature_selection_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:
```
static, context_switching_calls
```

Implements:

`public` *feature_scoring_protocol*

Imports:

`public` *feature_scoring_common*

Remarks:

(none)

Inherited public predicates:

score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.86.2 `chi_square_score`

Scores categorical feature values against categorical targets using the Pearson contingency-table chi-square statistic. Numeric values are treated as category labels.

Availability:

`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:

`static, context_switching_calls`

Extends:

`public` *chi_square_score(categorical)*

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.86.3 `chi_square_score(Discretization)`

- `Discretization` - The atom categorical, or equal_width(Count) or equal_frequency(Count) with a positive integer bin count.

Scores a categorical or explicitly discretized numeric feature against categorical targets using the Pearson contingency-table chi-square statistic, without correction or normalization.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *feature_scoring_protocol*
Imports:
`public` *feature_scoring_common*

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.86.4 `chi_square_yates_score`

Scores categorical features using Yates-corrected chi-square for occupied two-by-two tables and Pearson chi-square otherwise.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *feature_scoring_protocol*
Imports:
`public` *feature_scoring_common*

Remarks:
(none)

Inherited public predicates:

score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.86.5 `correlation_score`

Scores a numeric feature against a numeric target using squared Pearson correlation over complete examples. Returns a score in [0.0, 1.0], independent of correlation sign. Returns 0.0 for fewer than two complete examples or an exactly constant feature or target.

Availability:

```
logtalk_load(feature_selection_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *feature_scoring_protocol*

Imports:

`public` *feature_scoring_common*

Uses:

list

Remarks:

(none)

Inherited public predicates:

score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.86.6 cramers_v_bias_corrected_score

Scores categorical feature values using bias-corrected Cramer's V in [0.0, 1.0]. Numeric inputs are category labels; nonpositive corrected dimensions score zero.

Availability:

```
logtalk_load(feature_selection_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-08

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *cramers_v_bias_corrected_score(categorical)*

Remarks:

(none)

Inherited public predicates:

score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.86.7 `cramers_v_bias_corrected_score(Discretization)`

- `Discretization` - The atom categorical, or equal_width(Count) or equal_frequency(Count) with a positive integer bin count.

Scores categorical or explicitly discretized numeric features using bias-corrected Cramer's V in [0.0, 1.0]. Degenerate tables and nonpositive corrected dimensions score zero.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *feature_scoring_protocol*
Imports:
`public` *feature_scoring_common*

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.86.8 `cramers_v_score`

Scores categorical feature values against categorical targets using uncorrected Cramer's V in [0.0, 1.0]. Numeric values are treated as category labels.

Availability:
```
logtalk_load(feature_selection_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
```
static, context_switching_calls
```

Extends:
`public` *cramers_v_score(categorical)*

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.86.9 `cramers_v_score(Discretization)`

- `Discretization` - The atom categorical, or equal_width(Count) or equal_frequency(Count) with a positive integer bin count.

Scores a categorical or explicitly discretized numeric feature against categorical targets using uncorrected Cramer's V in [0.0, 1.0].

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *feature_scoring_protocol*
Imports:
`public` *feature_scoring_common*

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.86.10 `feature_dataset_protocol`

Protocol for datasets used with feature selection algorithms. Generalizes classification_protocols::dataset_protocol to also support regression targets and unsupervised use (no target at all), reusing the same attribute_values/2 predicate for familiarity.

Availability:
```
logtalk_load(feature_selection_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-02

Compilation flags:
```
static
```

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

    - *attribute_values/2*
    - *example_count/1*
    - *example/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

attribute_values/2

Enumerates by backtracking the candidate features and their possible values. Each atomic feature name must be declared once. For discrete features, `Values` is a list of possible values. For continuous (numeric) features, `Values` is the atom `continuous`. Never includes the target.

Compilation flags:

```
static
```

Template:

```
attribute_values(Feature,Values)
```

Mode and number of proofs:

```
attribute_values(?atomic,-list(atomic)) - zero_or_more
attribute_values(?atomic,-atom) - zero_or_more
```

---

example_count/1

Returns the number of examples. The declared count must match the number of examples enumerated by example/3.

Compilation flags:

```
static
```

Template:

```
example_count(Count)
```

Mode and number of proofs:

```
example_count(-positive_integer) - one
```

`example/3`

Enumerates by backtracking the examples in the dataset. Each example has an `Id`, a list of `Feature-Value` pairs with distinct declared feature names, and a `Target` value. A missing feature is omitted or has an unbound value. `Target` is an atom for classification, a number for regression, or an unbound variable when unknown.

Compilation flags:
`static`

Template:
`example(Id,Features,Target)`
Mode and number of proofs:
`example(-integer,-list(pair),?term)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.86.11 `feature_discretization`

Typed preparation of categorical columns with per-feature or joint complete-case discretization.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static`

Extends:
`public` *feature_scoring_common*
Uses:
avltree
list
type

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
  - *valid_feature_discretization/1*
  - *prepare_feature_columns/7*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

valid_feature_discretization/1

True for a ground categorical, equal-width, or equal-frequency specification with a positive integer bin count.

Compilation flags:

`static`

Template:

`valid_feature_discretization(Specification)`

Mode and number of proofs:

`valid_feature_discretization(@term)` - `zero_or_one`

---

prepare_feature_columns/7

Prepares `column(Feature,Specification,Pairs,CategoryCount)` terms in declaration order. Mode is `per_feature` or `joint`; fitting uses only complete observations.

Compilation flags:

`static`

Template:

`prepare_feature_columns(Dataset,Features,Examples,Options,Mode,Columns,Diagnostics)`

Mode and number of proofs:

`prepare_feature_columns(+object_identifier,+list(atomic),+list(compound),+list(compound),+atom,-list(compound),-list(compound))` - `one_or_error`

Exceptions:

A feature declaration is unsupported:
`domain_error(feature_type,Feature-Declaration)`
A categorical feature value is outside its declared domain:
`domain_error(feature_value,Feature-Value)`
An override names an undeclared feature:
`domain_error(unknown_feature,Feature)`
The preparation mode is unsupported:
`domain_error(preparation_mode,Mode)`
A continuous feature value is not numeric:
`type_error(number,Value)`
A categorical value or target is not atomic:
`type_error(atomic,Value)`
A declaration or configuration is unbound:
`instantiation_error`
A declaration domain is not a list:
`type_error(list,Domain)`
A binning specification is invalid:
`domain_error(discretization,Specification)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.86.12 `feature_redundancy`

Mutual information redundancy in bits between aligned, already prepared categorical feature columns.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static`

Extends:
`public` *feature_discretization*

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *feature_pair_mutual_information/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

feature_pair_mutual_information/3

Computes raw mutual information between the feature values in two same-length, complete `Value-Target` columns without refitting discretization.

Compilation flags:
`static`

Template:
`feature_pair_mutual_information(LeftPairs,RightPairs,Information)`
Mode and number of proofs:
`feature_pair_mutual_information(+list(pair),+list(pair),-float)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.86.13 feature_scoring_common

Shared arithmetic, discretization, and sparse contingency helpers for feature scoring.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:
`static`

Uses:
avltree
list
numberlist
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - `class_scatter/5`
  - `categorical_pairs/4`
  - `contingency_counts/2`
  - `contingency_min_expected_count/2`
  - `contingency_score/3`
  - `complete_pairs/3`
  - `complete_values/2`
  - `mean_values/2`
  - `variance_values/2`
  - `centered_values/2`
  - `scale_values/2`
  - `scale_values/3`
  - `squared_deviations/3`
  - `sum_of_squared_deviations/3`
  - `center/3`
  - `dot_product/3`
  - `sum_of_squares_list/2`
  - `split_pairs/3`
  - `group_by_target/2`
- *Private predicates*
  - `compare_numeric_pairs/3`
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

class_scatter/5

Computes scaled between-class and within-class scatter from complete numeric-value and categorical-target pairs. Empty inputs have zero scatter.

Compilation flags:

```
static
```

Template:

```
class_scatter(Pairs,ObservationCount,ClassCount,Between,Within)
```

Mode and number of proofs:

```
class_scatter(+list(pair),-non_negative_integer,-non_negative_integer,-float,
-float) - one
```

---

categorical_pairs/4

Validates aligned scoring inputs and prepares complete categorical pairs, optionally discretizing numeric features.

Compilation flags:

```
static
```

Template:

```
categorical_pairs(Discretization,Values,Targets,Pairs)
```

Mode and number of proofs:

```
categorical_pairs(+term,+list,+list,-list(pair)) - one_or_error
```

Exceptions:

An input list or discretization parameter is unbound:

```
instantiation_error
```

An input is not a list:

```
type_error(list,Input)
```

A categorical value or target is not atomic:

```
type_error(atomic,Value)
```

A continuous feature value is not numeric:

```
type_error(number,Value)
```

A bin count is not an integer:

```
type_error(integer,Count)
```

A bin count is not positive:

```
domain_error(positive_integer,Count)
```

The discretization specification is invalid:

```
domain_error(discretization,Discretization)
```

The input list lengths differ:

```
consistency_error(list_length,ValueCount,TargetCount)
```

---

contingency_counts/2

Builds sparse observed cell counts and marginal dictionaries from complete categorical pairs.

Compilation flags:

```
static
```

Template:

```
contingency_counts(Pairs,Counts)
```

Mode and number of proofs:

```
contingency_counts(+list(pair),-compound) - one
```

---

contingency_min_expected_count/2

Returns the minimum expectation over occupied marginal categories, including unobserved cells. Empty tables return 0.0.

Compilation flags:

```
static
```

Template:

```
contingency_min_expected_count(Counts,Minimum)
```

Mode and number of proofs:

```
contingency_min_expected_count(+compound,-float) - one
```

---

contingency_score/3

Computes mutual information, raw or Yates-corrected chi-square, symmetrical uncertainty, or uncorrected or bias-corrected Cramer's V from sparse contingency counts. Degenerate tables score 0.0.

Compilation flags:

```
static
```

Template:

```
contingency_score(Criterion,Counts,Score)
```

Mode and number of proofs:

```
contingency_score(+atom,+compound,-float) - one
```

---

complete_pairs/3

Collects the Value-Target pairs for which both the feature value and the target are bound, discarding any example where either is an unbound variable (casewise deletion). Values and Targets must have the same length.

Compilation flags:

```
static
```

Template:

```
complete_pairs(Values,Targets,Pairs)
```

Mode and number of proofs:

```
complete_pairs(+list,+list,-list(pair)) - one
```

---

complete_values/2

Collects the bound (non-missing) values of a list, discarding unbound ones.

Compilation flags:

```
static
```

Template:

```
complete_values(Values,Complete)
```

Mode and number of proofs:

```
complete_values(+list,-list(number)) - one
```

---

mean_values/2

Computes the arithmetic mean of a non-empty numeric list.

Compilation flags:

```
static
```

Template:

```
mean_values(Values,Mean)
```

Mode and number of proofs:

```
mean_values(+list(number),-float) - one_or_error
```

Exceptions:

Values is empty:

```
evaluation_error(zero_divisor)
```

---

variance_values/2

Computes the population variance of a non-empty numeric list.

Compilation flags:
`static`

Template:
`variance_values(Values,Variance)`
Mode and number of proofs:
`variance_values(+list(number),-float)` - `one_or_error`

Exceptions:
`Values` is empty:
`evaluation_error(zero_divisor)`

---

centered_values/2

Centers non-empty numeric values using offsets from the first value, preserving exact constant inputs.

Compilation flags:
`static`

Template:
`centered_values(Values,Centered)`
Mode and number of proofs:
`centered_values(+list(number),-list(number))` - `one_or_error`

Exceptions:
`Values` is empty:
`evaluation_error(zero_divisor)`

---

scale_values/2

Scales numeric values by their largest absolute value, leaving an all-zero list unchanged.

Compilation flags:
`static`

Template:
`scale_values(Values,Scaled)`
Mode and number of proofs:
`scale_values(+list(number),-list(number))` - `one`

---

scale_values/3

Scales numeric values and returns the divisor, using 1.0 for an all-zero or empty list.

Compilation flags:

```
static
```

Template:

```
scale_values(Values,Scaled,Scale)
```

Mode and number of proofs:

```
scale_values(+list(number),-list(number),-number) - one
```

---

squared_deviations/3

Computes the squared deviation of every value from a given reference value (typically the mean).

Compilation flags:

```
static
```

Template:

```
squared_deviations(Values,Reference,SquaredDeviations)
```

Mode and number of proofs:

```
squared_deviations(+list(number),+number,-list(number)) - one
```

---

sum_of_squared_deviations/3

Computes the sum of the squared deviations of every value from a given reference value (typically the mean); the within-group sum of squares used by anova_f_score is the sum of this over every group.

Compilation flags:

```
static
```

Template:

```
sum_of_squared_deviations(Values,Reference,SumSquaredDeviations)
```

Mode and number of proofs:

```
sum_of_squared_deviations(+list(number),+number,-float) - one
```

---

center/3

Subtracts a value (typically the mean) from every element of a numeric list.

Compilation flags:
```
static
```

Template:
```
center(Values,Value,Centered)
```
Mode and number of proofs:
```
center(+list(number),+number,-list(number)) - one
```

---

dot_product/3

Computes the dot product of two position-aligned numeric lists of the same length.

Compilation flags:
```
static
```

Template:
```
dot_product(Values1,Values2,Dot)
```
Mode and number of proofs:
```
dot_product(+list(number),+list(number),-number) - one
```

---

sum_of_squares_list/2

Computes the sum of the squares of a numeric list.

Compilation flags:
```
static
```

Template:
```
sum_of_squares_list(Values,SumSquares)
```
Mode and number of proofs:
```
sum_of_squares_list(+list(number),-number) - one
```

---

split_pairs/3

Splits a list of `Value1-Value2` pairs into separate, position-aligned lists of first and second values.

Compilation flags:
`static`

Template:
`split_pairs(Pairs,Values1,Values2)`
Mode and number of proofs:
`split_pairs(+list(pair),-list(number),-list(number))` - `one`

---

group_by_target/2

Groups `Value-Target` pairs into `Target-Values` pairs in standard target order, preserving input value order within each group.

Compilation flags:
`static`

Template:
`group_by_target(Pairs,Groups)`
Mode and number of proofs:
`group_by_target(+list(pair),-list(pair))` - `one`

Private predicates

compare_numeric_pairs/3

Compares numeric feature values arithmetically, keeping numerically equal representations together.

Compilation flags:
`static`

Template:
`compare_numeric_pairs(Order,Pair1,Pair2)`
Mode and number of proofs:
`compare_numeric_pairs(-atom,+pair,+pair)` - `one`

Operators

(none)
protocol

1.86.14 feature_scoring_protocol

Protocol for pluggable feature scoring criteria used by filter-based feature selection.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *score/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

score/3

Computes a relevance score from aligned feature values and targets in dataset order. Excludes unbound feature values and, for supervised criteria, unbound targets. Unsupervised criteria ignore targets. Higher scores indicate greater relevance, but score scales and thresholds are criterion-specific.

Compilation flags:
`static`

Template:
`score(Values,Targets,Score)`
Mode and number of proofs:
`score(+list,+list,-number)` - `one_or_error`

Exceptions:

A checked scoring input list or discretization parameter is unbound:

```
instantiation_error
```

A checked scoring input is not a list:

```
type_error(list,Input)
```

A complete categorical value or target is not atomic:

```
type_error(atomic,Value)
```

A complete feature value required to be numeric is not numeric:

```
type_error(number,Value)
```

A discretization bin count is not an integer:

```
type_error(integer,Count)
```

A discretization bin count is not positive:

```
domain_error(positive_integer,Count)
```

A discretization specification is invalid:

```
domain_error(discretization,Discretization)
```

Checked scoring input list lengths differ:

```
consistency_error(list_length,ValueCount,TargetCount)
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

category

1.86.15 `feature_selector_common`

Shared predicates for selector diagnostics, dataset validation, feature-matrix utilities, and selection strategies (top-k and threshold).

Availability:

```
logtalk_load(feature_selection_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:

```
static
```

Implements:

```
public feature_selector_protocol
```

Extends:

`public` *options*

Uses:

avltree
format
list
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_selector/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *feature_scores/2* *learn/2* *learn/3* *option/2* *option/3* *print_selector/1* *selected_features/2* *selector_options/2* *valid_option/1* *valid_options/1* *valid_selector/1*

- *Public predicates*
- *Protected predicates*
  - *selector_diagnostics_data/2*
  - *selector_export_template/4*
  - *selector_term_template/2*
  - *print_selector_template/1*
  - *dataset_examples/2*
  - *dataset_examples/3*
  - *feature_names/2*
  - *feature_values/3*
  - *targets/2*
  - *score_features/4*
  - *sort_by_decreasing_score/2*
  - *select_top_k/3*
  - *select_above_threshold/3*
  - *base_selector_diagnostics/5*
  - *valid_selector_metadata/2*
  - *valid_selector_metadata/3*
  - *check_scoring_metric/1*
- *Private predicates*
  - *compare_feature_scores/3*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

selector_diagnostics_data/2

Hook predicate that importing selector implementations must define in order to expose diagnostics metadata. A default implementation is provided that assumes the diagnostics list is the last argument of the selector term; concrete implementations following that convention do not need to override it.

Compilation flags:

```
static
```

Template:

```
selector_diagnostics_data(Selector,Diagnostics)
```

Mode and number of proofs:

```
selector_diagnostics_data(+compound,-list(compound)) - one
```

---

selector_export_template/4

Hook predicate that importing selector implementations must define in order to expose the exported selector template for a given functor.

Compilation flags:

```
static
```

Template:

```
selector_export_template(Dataset,Selector,Functor,Template)
```

Mode and number of proofs:

```
selector_export_template(+object_identifier,+compound,+atom,-callable) - one
```

---

selector_term_template/2

Hook predicate that importing selector implementations must define in order to expose the learned selector term template used by pretty-printing helpers.

Compilation flags:

```
static
```

Template:

```
selector_term_template(Selector,Template)
```

Mode and number of proofs:

```
selector_term_template(+compound,-callable) - one
```

---

print_selector_template/1

Pretty-printing helper predicate used by importing selector implementations to show the learned selector term template.

Compilation flags:
`static`

Template:
`print_selector_template(Selector)`
Mode and number of proofs:
`print_selector_template(+compound)` - `one`

---

dataset_examples/2

Collects and validates dataset examples. Rejects duplicate declarations, repeated features within examples, unknown features, and inconsistent example counts.

Compilation flags:
`static`

Template:
`dataset_examples(Dataset,Examples)`
Mode and number of proofs:
`dataset_examples(+object_identifier,-list(compound))` - `one_or_error`

Exceptions:
A feature name, feature list, or declared count is a variable:
`instantiation_error`
The dataset contains no examples:
`domain_error(non_empty_examples,Dataset)`
An example names a feature not declared by attribute_values/2:
`domain_error(unknown_feature,Feature)`
A feature is declared or supplied more than once:
`domain_error(duplicate_feature,Feature)`
A feature name is not atomic:
`type_error(atomic,Feature)`
An example feature list is not a list:
`type_error(list,Features)`
An example feature entry is not a pair:
`type_error(pair,Entry)`
The declared count is not an integer:
`type_error(integer,DeclaredCount)`
The declared count is not positive:
`domain_error(positive_integer,DeclaredCount)`
The declared and observed example counts differ:
`consistency_error(example_count,DeclaredCount,ObservedCount)`

---

dataset_examples/3

Collects declared feature names and validated examples in one pass over the dataset predicates. Uses the same validation as dataset_examples/2.

Compilation flags:

```
static
```

Template:

```
dataset_examples(Dataset,Features,Examples)
```

Mode and number of proofs:

```
dataset_examples(+object_identifier,-list(atomic),-list(compound)) - one_or_error
```

Exceptions:

A feature name, feature list, or declared count is a variable:

```
instantiation_error
```

The dataset contains no examples:

```
domain_error(non_empty_examples,Dataset)
```

A feature is declared or supplied more than once:

```
domain_error(duplicate_feature,Feature)
```

An example names an undeclared feature:

```
domain_error(unknown_feature,Feature)
```

A feature name is not atomic:

```
type_error(atomic,Feature)
```

An example feature list is not a list:

```
type_error(list,Features)
```

An example feature entry is not a pair:

```
type_error(pair,Entry)
```

The declared count is not an integer:

```
type_error(integer,DeclaredCount)
```

The declared count is not positive:

```
domain_error(positive_integer,DeclaredCount)
```

The declared and observed example counts differ:

```
consistency_error(example_count,DeclaredCount,ObservedCount)
```

---

feature_names/2

Returns unique, atomic candidate features in declaration order, rejecting invalid or duplicate declarations.

Compilation flags:

```
static
```

Template:

```
feature_names(Dataset,FeatureNames)
```

Mode and number of proofs:

```
feature_names(+object_identifier,-list(atomic)) - one_or_error
```

Exceptions:

A declared feature name is a variable:

```
instantiation_error
```

A declared feature name is not atomic:

```
type_error(atomic,Feature)
```

A feature is declared more than once:

```
domain_error(duplicate_feature,Feature)
```

---

feature_values/3

Returns the value of one feature for every example, in example order. An example that does not mention the feature contributes an unbound variable (a missing value).

Compilation flags:

```
static
```

Template:

```
feature_values(Examples,Feature,Values)
```

Mode and number of proofs:

```
feature_values(+list(compound),+atomic,-list) - one
```

---

targets/2

Returns the target of every example, in example order (an unbound variable for an example with no known target).

Compilation flags:

```
static
```

Template:

```
targets(Examples,Targets)
```

Mode and number of proofs:

```
targets(+list(compound),-list) - one
```

---

score_features/4

Scores every feature using score/3 messages to the metric object. Returns `Feature-Score` pairs in decreasing score order, preserving input feature order for numeric ties.

Compilation flags:

```
static
```

Template:

```
score_features(Metric,Examples,FeatureNames,FeatureScores)
```

Mode and number of proofs:

```
score_features(+object_identifier,+list(compound),+list(atomic),-list(pair)) - one
```

---

sort_by_decreasing_score/2

Sorts feature scores numerically in decreasing order, preserving input order for ties.

Compilation flags:

```
static
```

Template:

```
sort_by_decreasing_score(FeatureScores,Sorted)
```

Mode and number of proofs:

```
sort_by_decreasing_score(+list(pair),-list(pair)) - one
```

---

select_top_k/3

Returns the feature names of the K highest-scoring features (`FeatureScores` already sorted by decreasing score, as returned by score_features/4), preserving that order. Returns every feature name, still in that order, when fewer than K are given.

Compilation flags:

```
static
```

Template:

```
select_top_k(FeatureScores,K,Selected)
```

Mode and number of proofs:

```
select_top_k(+list(pair),+positive_integer,-list(atomic)) - one
```

---

select_above_threshold/3

Returns the feature names of every feature whose score is at least Threshold (`FeatureScores` already sorted by decreasing score, as returned by score_features/4), preserving that order.

Compilation flags:

```
static
```

Template:

```
select_above_threshold(FeatureScores,Threshold,Selected)
```

Mode and number of proofs:

```
select_above_threshold(+list(pair),+number,-list(atomic)) - one
```

---

base_selector_diagnostics/5

Builds the common part of a selector diagnostics metadata list, combined with implementation-specific extra diagnostics terms.

Compilation flags:

```
static
```

Template:

```
base_selector_diagnostics(Model,ExampleCount,Options,ExtraDiagnostics,
Diagnostics)
```

Mode and number of proofs:

```
base_selector_diagnostics(+atom,+positive_integer,+list(compound),
+list(compound),-list(compound)) - one
```

---

valid_selector_metadata/2

True when ground diagnostics metadata contains the expected model, a positive example count, and a list of stored options.

Compilation flags:

```
static
```

Template:

```
valid_selector_metadata(Model,Diagnostics)
```

Mode and number of proofs:

```
valid_selector_metadata(+atom,+list(compound)) - zero_or_one
```

---

valid_selector_metadata/3

True when diagnostics metadata contains the expected model term and records the given effective options.

Compilation flags:

```
static
```

Template:

```
valid_selector_metadata(Model,Options,Diagnostics)
```

Mode and number of proofs:

```
valid_selector_metadata(+atom,+list(compound),+list(compound)) - zero_or_one
```

---

check_scoring_metric/1

Checks that a scoring metric option value names an object implementing score/3 (feature_scoring_protocol).

Compilation flags:
`static`

Template:
`check_scoring_metric(Metric)`
Mode and number of proofs:
`check_scoring_metric(@callable)` - `one_or_error`

Exceptions:
`Metric` is a variable:
`instantiation_error`
`Metric` is bound but does not implement score/3:
`domain_error(scoring_metric,Metric)`

Private predicates

compare_feature_scores/3

Compares numerically decreasing scores, breaking ties by increasing input position.

Compilation flags:
`static`

Template:
`compare_feature_scores(Order,ScoredFeature1,ScoredFeature2)`
Mode and number of proofs:
`compare_feature_scores(-atom,+compound,+compound)` - `one`

Operators

(none)
protocol

1.86.16 feature_selector_protocol

Protocol for feature selection models.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-10-06

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/3*
  - *learn/2*
  - *selected_features/2*
  - *feature_scores/2*
  - *check_selector/1*
  - *valid_selector/1*
  - *diagnostics/2*
  - *diagnostic/2*
  - *selector_options/2*
  - *export_to_clauses/4*
  - *export_to_file/4*
  - *print_selector/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/3

Learns a feature selector from the given dataset object using the specified options.

Compilation flags:
`static`

Template:
`learn(Dataset,Selector,Options)`

Mode and number of proofs:
`learn(+object_identifier,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

A chi-square expectation is below the requested minimum:

`domain_error(chi_square_expected_count,Feature-expected(ObservedMinimum, RequiredMinimum))`

A generated regularization bound exceeds backend floating-point limits:

`evaluation_error(float_overflow)`

A feature declaration is unsupported by the receiving selector:

`domain_error(feature_type,Feature-Declaration)`

A complete feature value required to be numeric is not numeric:

`type_error(number,Value)`

A complete categorical target is not atomic:

`type_error(atomic,Target)`

A dataset feature name, feature list, or declared count is a variable:

`instantiation_error`

A dataset feature name is not atomic:

`type_error(atomic,Feature)`

An example feature list is not a list:

`type_error(list,Features)`

An example feature entry is not a pair:

`type_error(pair,Entry)`

The declared count is not an integer:

`type_error(integer,DeclaredCount)`

The declared count is not positive:

`domain_error(positive_integer,DeclaredCount)`

A feature is declared or supplied more than once:

`domain_error(duplicate_feature,Feature)`

An example names an undeclared feature:

`domain_error(unknown_feature,Feature)`

The dataset contains no examples:

`domain_error(non_empty_examples,Dataset)`

The declared and observed example counts differ:

`consistency_error(example_count,DeclaredCount,ObservedCount)`

learn/2

Learns a feature selector from the given dataset object using default options.

Compilation flags:

`static`

Template:

`learn(Dataset,Selector)`

Mode and number of proofs:

`learn(+object_identifier,-compound)` - `one_or_error`

Exceptions:

A feature declaration is unsupported by the receiving selector:

`domain_error(feature_type,Feature-Declaration)`

A complete feature value required to be numeric is not numeric:

`type_error(number,Value)`

A complete categorical target is not atomic:

`type_error(atomic,Target)`

A dataset feature name, feature list, or declared count is a variable:

`instantiation_error`

A dataset feature name is not atomic:

`type_error(atomic,Feature)`

An example feature list is not a list:

`type_error(list,Features)`

An example feature entry is not a pair:

`type_error(pair,Entry)`

The declared count is not an integer:

`type_error(integer,DeclaredCount)`

The declared count is not positive:

`domain_error(positive_integer,DeclaredCount)`

A feature is declared or supplied more than once:

`domain_error(duplicate_feature,Feature)`

An example names an undeclared feature:

`domain_error(unknown_feature,Feature)`

The dataset contains no examples:

`domain_error(non_empty_examples,Dataset)`

The declared and observed example counts differ:

`consistency_error(example_count,DeclaredCount,ObservedCount)`

selected_features/2

Returns the features selected by the learned selector.

Compilation flags:
`static`

Template:
`selected_features(Selector,Features)`
Mode and number of proofs:
`selected_features(+compound,-list(atomic))` - `one`

---

feature_scores/2

Returns every candidate feature with its relevance score, as a list of `Feature-Score` pairs sorted by decreasing score (not just the selected ones; see selected_features/2). Shared filter implementations preserve declaration order for numeric ties.

Compilation flags:
`static`

Template:
`feature_scores(Selector,FeatureScores)`
Mode and number of proofs:
`feature_scores(+compound,-list(pair))` - `one`

---

check_selector/1

Checks that a learned selector term is structurally valid for the receiving implementation. Throws an exception when the term is not a valid selector representation.

Compilation flags:
`static`

Template:
`check_selector(Selector)`
Mode and number of proofs:
`check_selector(@compound)` - `one_or_error`

Exceptions:
`Selector` is a variable:
`instantiation_error`
`Selector` is neither a variable nor a valid selector:
`domain_error(selector,Selector)`

---

valid_selector/1

True when a learned selector term is structurally valid for the receiving implementation. Succeeds iff check_selector/1 succeeds without throwing an exception.

Compilation flags:
`static`

Template:
`valid_selector(Selector)`
Mode and number of proofs:
`valid_selector(@compound)` - `zero_or_one`

---

diagnostics/2

Returns diagnostics metadata for a learned selector.

Compilation flags:
`static`

Template:
`diagnostics(Selector,Diagnostics)`
Mode and number of proofs:
`diagnostics(+compound,-list(compound))` - `one`

---

diagnostic/2

Enumerates individual diagnostics metadata terms for a learned selector.

Compilation flags:
`static`

Template:
`diagnostic(Selector,Diagnostic)`
Mode and number of proofs:
`diagnostic(+compound,?compound)` - `zero_or_more`

---

selector_options/2

Returns the effective training options recorded in a learned selector diagnostics metadata.

Compilation flags:
```
static
```

Template:
```
selector_options(Selector,Options)
```
Mode and number of proofs:
```
selector_options(+compound,-list(compound)) - one
```

---

export_to_clauses/4

Converts a selector into a list of predicate clauses. `Functor` is the functor for the generated predicate clauses.

Compilation flags:
```
static
```

Template:
```
export_to_clauses(Dataset,Selector,Functor,Clauses)
```
Mode and number of proofs:
```
export_to_clauses(+object_identifier,+compound,+atom,-list(clause)) - one
```

---

export_to_file/4

Exports a selector to a file. `Functor` is the functor for the generated predicate clauses.

Compilation flags:
```
static
```

Template:
```
export_to_file(Dataset,Selector,Functor,File)
```
Mode and number of proofs:
```
export_to_file(+object_identifier,+compound,+atom,+atom) - one
```

`print_selector/1`

Prints a selector to the current output stream in a human-readable format.

Compilation flags:
`static`

Template:
`print_selector(Selector)`
Mode and number of proofs:
`print_selector(+compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

feature_dataset_protocol, *feature_scoring_protocol*

category

1.86.17 `filter_feature_selector_common`

Shared learning, validation, selection, and export implementations for univariate feature selectors.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static`

Extends:
`public` *feature_selector_common*
Uses:
avltree
list

type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_selector/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *feature_scores/2* *learn/2* *learn/3* *option/2* *option/3* *print_selector/1* *selected_features/2* *selector_options/2* *valid_option/1* *valid_options/1* *valid_selector/1*

- *Public predicates*
- *Protected predicates*
  - *filter_model/1*
  - *filter_scoring_metric/2*
  - *filter_validate_dataset/3*
  - *filter_feature_scores/6*
  - *filter_validate_diagnostics/2*
  - *filter_valid_preparation_diagnostics/2*
  - *filter_selection/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

filter_model/1

Returns the receiving filter implementation model name.

Compilation flags:
`static`

Template:
`filter_model(Model)`
Mode and number of proofs:
`filter_model(-atom)` - `one`

filter_scoring_metric/2

Returns the scoring metric for the effective options.

Compilation flags:
`static`

Template:
`filter_scoring_metric(Options,Metric)`
Mode and number of proofs:
`filter_scoring_metric(+list(compound),-object_identifier)` - `one`

---

filter_validate_dataset/3

Checks algorithm-specific feature declarations. The default accepts all declarations.

Compilation flags:
`static`

Template:
`filter_validate_dataset(Dataset,Features,Examples)`
Mode and number of proofs:
`filter_validate_dataset(+object_identifier,+list(atomic),+list(compound))` - `one_or_error`

Exceptions:
A feature declaration is unsupported by the receiving filter:
`domain_error(feature_type,Feature-Declaration)`

---

filter_feature_scores/6

Scores all features and reports complete-case counts. Implementations can override preprocessing and scoring.

Compilation flags:
`static`

Template:
`filter_feature_scores(Dataset,Features,Examples,Options,Scores,Diagnostics)`
Mode and number of proofs:
`filter_feature_scores(+object_identifier,+list(atomic),+list(compound),+list(compound),-list(pair),-list(compound))` - `one_or_error`

Exceptions:
A complete numeric feature value is not numeric:
`type_error(number,Value)`

A complete categorical target is not atomic:

```
type_error(atomic,Target)
```

A chi-square expectation is below the requested minimum:

```
domain_error(chi_square_expected_count,Feature-expected(ObservedMinimum,
RequiredMinimum))
```

---

filter_validate_diagnostics/2

Checks implementation-specific preparation metadata against effective options.

Compilation flags:

```
static
```

Template:

```
filter_validate_diagnostics(Options,Diagnostics)
```

Mode and number of proofs:

```
filter_validate_diagnostics(+list(compound),+list(compound)) - zero_or_one
```

---

filter_valid_preparation_diagnostics/2

Checks the recorded preparation mode and joint complete-case counts.

Compilation flags:

```
static
```

Template:

```
filter_valid_preparation_diagnostics(Mode,Diagnostics)
```

Mode and number of proofs:

```
filter_valid_preparation_diagnostics(+atom,+list(compound)) - zero_or_one
```

---

filter_selection/3

Applies the receiving filter selection strategy to sorted feature scores.

Compilation flags:

```
static
```

Template:

```
filter_selection(Strategy,Scores,Selected)
```

Mode and number of proofs:

```
filter_selection(+term,+list(pair),-list(atomic)) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.86.18 `fisher_score`

Scores a numeric feature against categorical targets using between-class scatter divided by within-class scatter, capped at 1.0e10.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *feature_scoring_protocol*
Imports:
`public` *feature_scoring_common*
Uses:
list
type

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *check_complete_pairs/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`check_complete_pairs/2`

Checks complete numeric feature values and atomic target labels.

Compilation flags:
`static`

Template:
`check_complete_pairs(Pairs,Context)`
Mode and number of proofs:
`check_complete_pairs(+list(pair),+term)` - `one_or_error`

Exceptions:
A feature value is not numeric:
`type_error(number,Value)`
A target label is not atomic:
`type_error(atomic,Target)`

Operators

(none)
object

1.86.19 `mutual_information_score`

Scores categorical feature values against categorical targets using empirical mutual information in bits. Numeric values are treated as category labels.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:
`static, context_switching_calls`

Extends:

`public` *mutual_information_score(categorical)*

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.86.20 `mutual_information_score(Discretization)`

- `Discretization` - The atom categorical, or equal_width(Count) or equal_frequency(Count) with a positive integer bin count.

Scores a categorical or explicitly discretized numeric feature against categorical targets using empirical mutual information in bits.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *feature_scoring_protocol*
Imports:

`public` *feature_scoring_common*

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.86.21 `relief_feature_selector_common`

Shared joint nearest-neighbor feature selection for Relief, ReliefF, and RReliefF.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static`

Extends:
`public` *feature_selector_common*
Uses:
avltree
fast_random(Algorithm)

list
type

Remarks:
(none)

Inherited public predicates:
check_option/1 check_options/1 check_selector/1 default_option/1 default_options/1 diagnostic/2 diagnostics/2 export_to_clauses/4 export_to_file/4 feature_scores/2 learn/2 learn/3 option/2 option/3 print_selector/1 selected_features/2 selector_options/2 valid_option/1 valid_options/1 valid_selector/1

- *Public predicates*
- *Protected predicates*
  - `relief_model/1`
  - `relief_variant/1`
  - `anchors/4`
- *Private predicates*
  - `declarations/3`
  - `check_target/2`
  - `row_values/4`
  - `population/4`
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

`relief_model/1`

Returns the receiving selector model functor.

Compilation flags:
`static`

Template:
`relief_model(Model)`
Mode and number of proofs:
`relief_model(-atom)` - `one`

relief_variant/1

Returns `binary`, `multiclass`, or `regression`.

Compilation flags:
`static`

Template:
`relief_variant(Variant)`
Mode and number of proofs:
`relief_variant(-atom)` - `one`

---

anchors/4

Processes all rows or samples anchors while restoring RNG state on every exit path.

Compilation flags:
`static`

Template:
`anchors(Size,Seed,Rows,Anchors)`
Mode and number of proofs:
`anchors(+term,+positive_integer,+list(compound),-list(compound))` - `zero_or_one_or_error`

Exceptions:
A sampling size is not evaluable:
`type_error(evaluable,Size)`
Sampling arithmetic exceeds backend limits:
`evaluation_error(Reason)`

Private predicates

declarations/3

Resolves continuous and discrete feature declarations to numeric and categorical types.

Compilation flags:
`static`

Template:
`declarations(Features,Dataset,Declarations)`
Mode and number of proofs:
`declarations(+list(atomic),+object_identifier,-list(pair))` - `one_or_error`

Exceptions:

A declaration is unsupported:

```
domain_error(feature_type,Feature-Declaration)
```

---

check_target/2

Checks the target type required by the algorithm variant.

Compilation flags:

```
static
```

Template:

```
check_target(Variant,Target)
```

Mode and number of proofs:

```
check_target(+atom,+nonvar) - one_or_error
```

Exceptions:

A regression target is not numeric:

```
type_error(number,Target)
```

A classification target is not atomic:

```
type_error(atomic,Target)
```

---

row_values/4

Collects typed row values from an indexed feature dictionary without binding missing values.

Compilation flags:

```
static
```

Template:

```
row_values(Declarations,Dictionary,Values,Complete)
```

Mode and number of proofs:

```
row_values(+list(pair),+compound,-list,-boolean) - one_or_error
```

Exceptions:

A known numeric feature is not numeric:

```
type_error(number,Value)
```

A known categorical feature is not atomic:

```
type_error(atomic,Value)
```

A known categorical value is outside its declared domain:

```
domain_error(feature_value,Feature-Value)
```

---

population/4

Checks eligible class counts or regression row count and prepares target normalization.

Compilation flags:
`static`

Template:
`population(Variant,Rows,Classes,TargetScale)`
Mode and number of proofs:
`population(+atom,+list(compound),-list(pair),-term)` - `one_or_error`

Exceptions:
The eligible population does not satisfy the variant minimum:
`domain_error(relief_population,Variant-Counts)`

Operators

(none)
object

1.86.22 symmetrical_uncertainty_score

Scores categorical feature values against categorical targets using symmetrical uncertainty in [0.0, 1.0]. Numeric values are treated as category labels.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *symmetrical_uncertainty_score(categorical)*

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.86.23 `symmetrical_uncertainty_score(Discretization)`

- `Discretization` - The atom categorical, or equal_width(Count) or equal_frequency(Count) with a positive integer bin count.

Scores a categorical or explicitly discretized numeric feature against categorical targets using symmetrical uncertainty in [0.0, 1.0].

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *feature_scoring_protocol*
Imports:
`public` *feature_scoring_common*

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.86.24 `variance_score`

Unsupervised feature scoring by population variance: ignores Targets entirely. A near-constant feature (one that takes almost the same value in every example) carries little information for distinguishing examples and so scores low; this is the classic "variance threshold" filter criterion. Scores 0.0 when every value is missing.

Availability:
`logtalk_load(feature_selection_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *feature_scoring_protocol*
Imports:
`public` *feature_scoring_common*
Uses:
list

Remarks:
(none)

Inherited public predicates:
score/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.87 fisher_score_feature_selector

object

1.87.1 `fisher_score_feature_selector`

Selects continuous features for categorical targets using Fisher scores, top-k selection, or a score threshold.

Availability:
`logtalk_load(fisher_score_feature_selector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *filter_feature_selector_common*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_selector/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *feature_scores/2* *learn/2* *learn/3* *option/2* *option/3* *print_selector/1* *selected_features/2* *selector_options/2* *valid_option/1* *valid_options/1* *valid_selector/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *check_declarations/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

check_declarations/1

Checks that every feature is declared continuous.

Compilation flags:
`static`

Template:
`check_declarations(Declarations)`

Mode and number of proofs:
`check_declarations(+list(pair))` - `one_or_error`

Exceptions:
A feature is not declared continuous:
`domain_error(feature_type,Feature-Declaration)`

Operators

(none)

1.88 flags

category

1.88.1 flags

Implementation of persistent object flags.

Availability:
`logtalk_load(flags(loader))`

Author: Theofrastos Mantadelis
Version: 1:0:0
Date: 2010-11-27

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *get_flag_value/2*
  - *set_flag_value/2*
  - *set_flag_value/3*
  - *reset_flags/0*
  - *reset_flags/1*
  - *flag_groups/1*
  - *flag_group_chk/1*
  - *print_flags/0*
  - *print_flags/1*
  - *defined_flag/6*
  - *built_in_flag/2*
- *Protected predicates*
  - *unsafe_set_flag_value/2*
  - *define_flag/1*
  - *define_flag/2*
- *Private predicates*
  - *defined_flag_/6*
  - *flag_value_/2*
  - *validate/3*
  - *validate_type/1*
  - *is_validator/1*
- *Operators*

Public predicates

get_flag_value/2

Gets or tests the value of a flag.

Compilation flags:
```
static
```

Template:
```
get_flag_value(Flag,Value)
```
Mode and number of proofs:
```
get_flag_value(+atom,?nonvar) - zero_or_one
```

---

set_flag_value/2

Sets the value of a flag.

Compilation flags:
```
static
```

Template:
```
set_flag_value(Flag,NewValue)
```
Mode and number of proofs:
```
set_flag_value(+atom,@nonvar) - one
```

---

set_flag_value/3

Sets the value of a flag, returning the old value.

Compilation flags:
```
static
```

Template:
```
set_flag_value(Flag,OldValue,NewValue)
```
Mode and number of proofs:
```
set_flag_value(+atom,?nonvar,@nonvar) - one
```

---

reset_flags/0

Resets all flags to their default values.

Compilation flags:
`static`

Mode and number of proofs:
`reset_flags` - `one`

---

reset_flags/1

Resets all flags in a group to their default values.

Compilation flags:
`static`

Template:
`reset_flags(Group)`
Mode and number of proofs:
`reset_flags(+atom)` - `one`

---

flag_groups/1

Returns a list of all flag groups.

Compilation flags:
`static`

Template:
`flag_groups(Groups)`
Mode and number of proofs:
`flag_groups(-list(atom))` - `one`

---

flag_group_chk/1

Checks if a given atom is a flag group.

Compilation flags:
`static`

Template:

```
flag_group_chk(Group)
```
Mode and number of proofs:
```
flag_group_chk(+atom) - zero_or_one
```

---

print_flags/0

Prints a listing of all flags.

Compilation flags:
```
static
```

Mode and number of proofs:
```
print_flags - one
```

---

print_flags/1

Prints a listing of all flags in a group.

Compilation flags:
```
static
```

Template:
```
print_flags(Group)
```
Mode and number of proofs:
```
print_flags(+atom) - one
```

---

defined_flag/6

Gets or test the existing (visible) flag definitions.

Compilation flags:
```
static
```

Template:
```
defined_flag(Flag,Group,Type,DefaultValue,Description,Access)
```
Mode and number of proofs:
```
defined_flag(?atom,?atom,?nonvar,?nonvar,?atom,?atom) - zero_or_more
```

---

built_in_flag/2

True if the argument is a built-in flag type with the specified default value.

Compilation flags:
`static`

Template:
`built_in_flag(Type,DefaultValue)`
Mode and number of proofs:
`built_in_flag(?atom,?nonvar)` - `zero_or_more`

Protected predicates

unsafe_set_flag_value/2

Sets the value of a flag without performing any validation checks.

Compilation flags:
`static`

Template:
`unsafe_set_flag_value(Flag,NewValue)`
Mode and number of proofs:
`unsafe_set_flag_value(+atom,@nonvar)` - `one`

---

define_flag/1

Defines a new flag using default options.

Compilation flags:
`static`

Template:
`define_flag(Flag)`
Mode and number of proofs:
`define_flag(+atom)` - `one`

---

define_flag/2

Defines a new flag using a given set of options (for example, [group(general), type(nonvar), default(true), description(Flag), access(read_write)]).

Compilation flags:
`static`

Template:
`define_flag(Flag,Options)`
Mode and number of proofs:
`define_flag(+atom,@list)` - `one`

Private predicates

defined_flag_/6

Gets or test the existing flag definitions.

Compilation flags:
`dynamic`

Template:
`defined_flag_(Flag,Group,Type,DefaultValue,Description,Access)`
Mode and number of proofs:
`defined_flag_(?atom,?atom,?nonvar,?nonvar,?atom,?atom)` - `zero_or_more`

---

flag_value_/2

Table of flag values.

Compilation flags:
`dynamic`

Template:
`flag_value_(Flag,Value)`
Mode and number of proofs:
`flag_value_(?atom,?nonvar)` - `zero_or_more`

---

validate/3

Validates a flag value against its type, throwing an error identifying the flag if validation fails.

Compilation flags:
`static`

Template:
`validate(Type,Value,Flag)`
Mode and number of proofs:
`validate(+atom,@term,+atom)` - `zero_or_one`

---

validate_type/1

True if the argument is a valid flag type, either a built-in type or a custom validator object that validates itself.

Compilation flags:
`static`

Template:
`validate_type(Type)`
Mode and number of proofs:
`validate_type(+atom)` - `zero_or_one`

---

is_validator/1

True if the argument is a built-in flag type or an object implementing the flags_validator protocol.

Compilation flags:
`static`

Template:
`is_validator(Type)`
Mode and number of proofs:
`is_validator(+atom)` - `zero_or_one`

Operators

(none)
protocol

1.88.2 flags_validator

Flag validation protocol. Must be implemented by validator objects.

Availability:
`logtalk_load(flags(loader))`

Author: Theofrastos Mantadelis
Version: 1:1:0
Date: 2026-08-10

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *print_flags/0*
  - *validate/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

print_flags/0

Validates the validator object itself.

Compilation flags:
`static`

Mode and number of proofs:
`print_flags` - `zero_or_one`

validate/1

Validates a flag value.

Compilation flags:
`static`

Template:
`validate(Value)`
Mode and number of proofs:
`validate(@term)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.89 flow_networks

object

1.89.1 capacity_scaling

Capacity-scaling maximum flow using the `avltree` dictionary.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *capacity_scaling(avltree)*

Remarks:
(none)

Inherited public predicates:

max_flow/5 *min_cut/5* *min_cuts/5* *multi_source_max_flow/5* *node_capacity_max_flow/6* *sink_closest_min_cut/5* *unique_min_cut/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

capacity_scaling(Dictionary), *dinic*

object

1.89.2 `capacity_scaling(Dictionary)`

- `Dictionary` - Dictionary object used by the flow network.

Capacity-scaling augmenting paths. Each phase traverses residual arcs with capacity at least a power-of-two threshold. The standard bound is O(E^2*log(U)), before dictionary costs. This is an alternative to Dinic, not a guaranteed speedup.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *max_flow_common(Dictionary)*

Remarks:

(none)

Inherited public predicates:

max_flow/5 *min_cut/5* *min_cuts/5* *multi_source_max_flow/5* *node_capacity_max_flow/6* *sink_closest_min_cut/5* *unique_min_cut/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

capacity_scaling, *dinic(Dictionary)*, *edmonds_karp(Dictionary)*

object

1.89.3 `dinic`

Dinic maximum flow algorithm using the `avltree` dictionary.

Availability:

```
logtalk_load(flow_networks(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public dinic(avltree)
```

Remarks:
(none)

Inherited public predicates:
max_flow/5 *min_cut/5* *min_cuts/5* *multi_source_max_flow/5* *node_capacity_max_flow/6* *sink_closest_min_cut/5* *unique_min_cut/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dinic(Dictionary), *edmonds_karp*

object

1.89.4 `dinic(Dictionary)`

- `Dictionary` - Dictionary object used by the flow network (e.g., `avltree`).

Dinic maximum flow algorithm. Each phase builds a level graph using breadth-first search and then computes a blocking flow using depth-first search. Runs in O(V^2*E) time. A vertex whose depth-first search fails to push the requested amount is recorded as dead for the rest of the phase, which plays the role of the current-arc pointers of imperative implementations.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Imports:

`public` *max_flow_common(Dictionary)*

Remarks:

(none)

Inherited public predicates:

max_flow/5 *min_cut/5* *min_cuts/5* *multi_source_max_flow/5* *node_capacity_max_flow/6* *sink_closest_min_cut/5* *unique_min_cut/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

edmonds_karp(Dictionary), *dinic*, *flow_network(Dictionary)*

object

1.89.5 `edmonds_karp`

Edmonds-Karp maximum flow algorithm using the `avltree` dictionary.

Availability:

`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:

`static, context_switching_calls`

Extends:
`public` *edmonds_karp(avltree)*

Remarks:
(none)

Inherited public predicates:
max_flow/5 *min_cut/5* *min_cuts/5* *multi_source_max_flow/5* *node_capacity_max_flow/6* *sink_closest_min_cut/5* *unique_min_cut/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

edmonds_karp(Dictionary), *dinic*

object

1.89.6 `edmonds_karp(Dictionary)`

- `Dictionary` - Dictionary object used by the flow network (e.g., `avltree`).

Edmonds-Karp maximum flow algorithm (Ford-Fulkerson using shortest augmenting paths found by breadth-first search). Runs in O(V*E^2) time. Mainly intended as a simple reference implementation; see the `dinic` object for a faster alternative.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *max_flow_common(Dictionary)*

Remarks:
(none)

Inherited public predicates:
max_flow/5 *min_cut/5* *min_cuts/5* *multi_source_max_flow/5* *node_capacity_max_flow/6* *sink_closest_min_cut/5* *unique_min_cut/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dinic(Dictionary), *edmonds_karp*, *flow_network(Dictionary)*

object

1.89.7 `flow_network`

Flow network construction and inspection using the `avltree` dictionary.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *flow_network(avltree)*

Remarks:
(none)

Inherited public predicates:
arcs/2 *cut_capacity/3* *decompose_flow/3* *flow_balances/3* *flow_cost/3* *from_graph/2* *new/2* *new/3* *valid_cut/5* *valid_flow/3* *valid_min_cost_flow/4* *valid_min_cost_flow/5* *valid_st_flow/5* *vertices/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

flow_network(Dictionary)

object

1.89.8 `flow_network(Dictionary)`

- `Dictionary` - Dictionary object (e.g., `avltree`, `rbtree`, `bintree`, or `splaytree`).

Flow network construction and inspection. The network is an opaque term. Each input arc is stored as a residual pair of arcs: a forward arc with identifier `2*I` and a reverse arc with identifier `2*I+1` (`I` being the zero-based position of the arc in the input list), where the reverse arc has zero initial residual capacity and the negated cost. The parametric object parameter is the dictionary object used for the representation and must be the same one used by the flow algorithm objects.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *flow_network_protocol*
Uses:
list
type
weighted_directed_graph(Dictionary)

Remarks:
(none)

Inherited public predicates:
arcs/2 *cut_capacity/3* *decompose_flow/3* *flow_balances/3* *flow_cost/3* *from_graph/2* *new/2* *new/3* *valid_cut/5* *valid_flow/3* *valid_min_cost_flow/4* *valid_min_cost_flow/5* *valid_st_flow/5* *vertices/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

flow_network, *max_flow_protocol*, *min_cost_flow_protocol*

protocol

1.89.9 `flow_network_protocol`

Protocol for flow network construction and inspection. A flow network is a directed graph where each arc has a non-negative integer lower bound and capacity, with the lower bound not exceeding the capacity, and an integer cost.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *new/2*
  - *new/3*
  - *from_graph/2*

  - *vertices/2*
  - *arcs/2*
  - *flow_cost/3*
  - *flow_balances/3*
  - *cut_capacity/3*
  - *valid_cut/5*
  - *decompose_flow/3*
  - *valid_flow/3*
  - *valid_st_flow/5*
  - *valid_min_cost_flow/4*
  - *valid_min_cost_flow/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new/2

Constructs a flow network from a list of arcs. Arcs are `arc(From, To, Capacity)`, `arc(From, To, Capacity, Cost)`, or `arc(From, To, Lower, Capacity, Cost)` terms. The default cost and lower bound are zero. Vertices can be any ground term. Parallel, antiparallel, and self-loop arcs are allowed.

Compilation flags:

`static`

Template:

`new(Arcs,Network)`

Mode and number of proofs:

`new(+list(ground),--compound)` - `one_or_error`

Exceptions:

`Arcs` is a partial list or a list with an element `Element` which is not ground:

`instantiation_error`

`Arcs` is neither a variable nor a list:

`type_error(list(ground),Arcs)`

An element `Arc` of `Arcs` list is not an `arc/3`, `arc/4`, or `arc/5` term:

`domain_error(flow_arc,Arc)`

A capacity, lower bound or cost value is not an integer:

`type_error(integer,Value)`

A lower bound or capacity value is negative:

`domain_error(non_negative_integer,Value)`

A lower bound exceeds the capacity:

`consistency_error(arc_lower_bound_and_capacity,Lower,Capacity)`

new/3

As new/2, also preserving explicitly listed vertices, including isolated vertices. The vertex set is the sorted, duplicate-free union of Vertices and arc endpoints.

Compilation flags:

```
static
```

Template:

```
new(Vertices,Arcs,Network)
```

Mode and number of proofs:

```
new(+list(ground),+list(ground),--compound) - one_or_error
```

Exceptions:

Vertices is a partial list or a list with an element Element which is not ground:

```
instantiation_error
```

Vertices is neither a partial list nor a list:

```
type_error(list(ground),Vertices)
```

Arcs is a partial list or a list with an element Element which is not ground:

```
instantiation_error
```

Arcs is neither a variable nor a list:

```
type_error(list(ground),Arcs)
```

An element Arc of Arcs list is not an arc/3, arc/4, or arc/5 term:

```
domain_error(flow_arc,Arc)
```

A capacity, lower bound or cost value is not an integer:

```
type_error(integer,Value)
```

A lower bound or capacity value is negative:

```
domain_error(non_negative_integer,Value)
```

A lower bound exceeds the capacity:

```
consistency_error(arc_lower_bound_and_capacity,Lower,Capacity)
```

---

from_graph/2

Constructs a flow network from a weighted directed graph (created using the graphs library with the same dictionary object). Edge weights are interpreted as capacities and all costs are zero.

Compilation flags:

```
static
```

Template:

```
from_graph(Graph,Network)
```

Mode and number of proofs:

```
from_graph(+compound,--compound) - one_or_error
```

---

vertices/2

Returns the sorted list of vertices of a flow network (including vertices that only occur in self-loops).

Compilation flags:
`static`

Template:
`vertices(Network,Vertices)`
Mode and number of proofs:
`vertices(+compound,-list)` - `one`

---

arcs/2

Returns the arcs of a flow network as `arc(From, To, Lower, Capacity, Cost)` terms in the order they were given to `new/2`, including self-loops.

Compilation flags:
`static`

Template:
`arcs(Network,Arcs)`
Mode and number of proofs:
`arcs(+compound,-list(compound))` - `one`

---

flow_cost/3

Computes the cost of a capacity-valid flow, without requiring conservation or optimality. Each entry must be a `flow(ArcId, From, To, Amount)` term with a unique one-based arc ID, matching endpoints, and a positive integer amount. Every arc lower and upper bound must be respected, including omitted mandatory arcs. Entries may be in any order. Self-loop flows contribute their full cost.

Compilation flags:
`static`

Template:
`flow_cost(Network,Flow,Cost)`
Mode and number of proofs:
`flow_cost(+compound,+list(compound),-integer)` - `one_or_error`

Exceptions:
`Flow` is a variable or partial list, or an entry has a variable field:
`instantiation_error`
`Flow` is neither a variable nor a list:
`type_error(list,Flow)`

An entry is not a `flow/4` term:
```
domain_error(flow,Term)
```
An arc ID or amount is not an integer:
```
type_error(integer,Value)
```
An arc ID or amount is not positive:
```
domain_error(positive_integer,Value)
```
An arc ID is not in the network:
```
existence_error(arc,ArcId)
```
An entry has incorrect endpoints:
```
domain_error(flow_arc_endpoints,Term)
```
Arc IDs are repeated or an arc bound is violated:
```
domain_error(capacity_valid_flow,Flow)
```

---

flow_balances/3

Returns sorted `Vertex-NetOutflow` pairs for every vertex, including zero balances. Positive balances indicate supplies and negative balances indicate demands. Self-loops contribute zero net outflow. Requires a capacity-valid flow with the same validation and exceptions as `flow_cost/3`, but does not require conservation against particular supplies.

Compilation flags:
```
static
```

Template:
```
flow_balances(Network,Flow,Balances)
```
Mode and number of proofs:
```
flow_balances(+compound,+list(compound),-list(pair)) - one_or_error
```

---

cut_capacity/3

Computes the capacity of the cut whose source side is the given vertex set and whose sink side is its complement. Input order is immaterial; vertices must be unique and belong to the network. Empty and full sets are allowed. Counts parallel crossing arcs individually and excludes self-loops. Requires zero lower bounds.

Compilation flags:
```
static
```

Template:
```
cut_capacity(Network,SourceSide,Capacity)
```
Mode and number of proofs:
```
cut_capacity(+compound,+list(ground),-integer) - one_or_error
```

Exceptions:
`SourceSide` is a variable or partial list or contains a non-ground vertex:
```
instantiation_error
```
`SourceSide` is neither a variable nor a list:

`type_error(list,SourceSide)`

A vertex is not in the network:

`existence_error(vertex,Vertex)`

A vertex is repeated:

`domain_error(unique_cut_vertices,Vertex)`

The network contains a non-zero lower bound:

`domain_error(zero_lower_bounds,Network)`

---

valid_cut/5

Checks a ground `cut(SourceSide, SinkSide, Arcs)` candidate and its reported capacity. The sides must be sorted, duplicate-free lists forming a complete disjoint partition of the network vertices, with distinct existing source and sink on their respective sides. `Arcs` must list every positive-capacity crossing arc in network input order, including parallel arcs. Requires zero lower bounds and fails for invalid candidates. Cut validity does not imply minimum-cut optimality.

Compilation flags:

`static`

Template:

`valid_cut(Network,Source,Sink,Value,Cut)`

Mode and number of proofs:

`valid_cut(+compound,+ground,+ground,+integer,+compound)` - `zero_or_one`

---

decompose_flow/3

Decomposes a feasible flow into a list of `path(Vertices, ArcIds, Amount)` and `cycle(Vertices, ArcIds, Amount)` terms. Path endpoints are inferred from the flow balances. Vertex lists for cycles repeat the first vertex at the end. Arc IDs are one-based input arc positions.

Compilation flags:

`static`

Template:

`decompose_flow(Network,Flow,Decomposition)`

Mode and number of proofs:

`decompose_flow(+compound,+list(compound),-list(compound))` - `zero_or_one`

---

valid_flow/3

Checks that `Flow` respects every arc lower and upper capacity bound and that vertex net outflows match the given supplies and demands. Duplicate supply entries are added.

Compilation flags:

```
static
```

Template:

```
valid_flow(Network,Supplies,Flow)
```

Mode and number of proofs:

```
valid_flow(+compound,+list(pair),+list(compound)) - zero_or_one
```

---

valid_st_flow/5

Checks that `Flow` respects capacities and conservation for a source-to-sink flow of the given value. Fails if either terminal is not ground.

Compilation flags:

```
static
```

Template:

```
valid_st_flow(Network,Source,Sink,Value,Flow)
```

Mode and number of proofs:

```
valid_st_flow(+compound,+ground,+ground,+integer,+list(compound)) - zero_or_one
```

---

valid_min_cost_flow/4

Checks flow feasibility, the reported cost, and min-cost optimality by checking for negative-cost cycles in the residual graph.

Compilation flags:

```
static
```

Template:

```
valid_min_cost_flow(Network,Supplies,Flow,Cost)
```

Mode and number of proofs:

```
valid_min_cost_flow(+compound,+list(pair),+list(compound),+integer) - zero_or_one
```

`valid_min_cost_flow/5`

Checks flow feasibility and cost, and validates a potential certificate by verifying non-negative reduced costs on every positive-capacity residual arc.

Compilation flags:
`static`

Template:
`valid_min_cost_flow(Network,Supplies,Flow,Cost,Potentials)`

Mode and number of proofs:
`valid_min_cost_flow(+compound,+list(pair),+list(compound),+integer,+list(pair))` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

flow_network(Dictionary), *flow_network*, *max_flow_protocol*, *min_cost_flow_protocol*

category

1.89.10 `flow_support(Dictionary)`

- `Dictionary` - Dictionary object used by the flow network.

Support predicates shared by the flow algorithm objects: terminal validation, residual graph traversal, path augmentation, and flow extraction. The category parameter is the dictionary object and must be the same one used to construct the flow network.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static`

Uses:
list

queue

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *validate_terminals/3*
  - *partner/2*
  - *adjacency/3*
  - *bfs/5*
  - *bfs/6*
  - *augment/4*
  - *augment_arc/4*
  - *path_ids/4*
  - *bottleneck/3*
  - *flow_list/3*
  - *flow_cost/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

validate_terminals/3

Checks that the source and sink are distinct vertices of the network.

Compilation flags:
`static`

Template:
`validate_terminals(Network,Source,Sink)`
Mode and number of proofs:
`validate_terminals(+compound,+ground,+ground) - one_or_error`

Exceptions:
`Source` or `Sink` is not ground:
`instantiation_error`
`Source` or `Sink` is not a vertex of the network:
`existence_error(vertex,Vertex)`
`Source` and `Sink` are the same vertex:

```
consistency_error(distinct_vertices,Source,Sink)
```

---

partner/2

Arc identifier of the opposite arc in a residual pair.

Compilation flags:
`static`

Template:
`partner(Id,Partner)`
Mode and number of proofs:
`partner(+integer,-integer)` - `one`

---

adjacency/3

Outgoing residual arc identifiers of a vertex (empty list if none).

Compilation flags:
`static`

Template:
`adjacency(Vertex,Network,Ids)`
Mode and number of proofs:
`adjacency(+ground,+compound,-list(integer))` - `one`

---

bfs/5

Breadth-first search over arcs with positive residual capacity. `Stop` is either `stop(Vertex)` (stop as soon as the vertex is reached) or `all`. `Info` is a dictionary mapping each reached vertex to `Level-ArcId` where `ArcId` is the arc used to reach it (`none` for the source).

Compilation flags:
`static`

Template:
`bfs(Network,Residual,Source,Stop,Info)`
Mode and number of proofs:
`bfs(+compound,+compound,+ground,+nonvar,-compound)` - `one`

---

bfs/6

As `bfs/5`, traversing only arcs whose residual capacity is at least `Threshold`.

Compilation flags:
`static`

Template:
`bfs(Network,Residual,Source,Stop,Threshold,Info)`
Mode and number of proofs:
`bfs(+compound,+compound,+ground,+nonvar,+positive_integer,-compound)` - `one`

---

augment/4

Pushes the given amount of flow along a list of arc identifiers, updating the residual dictionary.

Compilation flags:
`static`

Template:
`augment(Ids,Amount,Residual0,Residual)`
Mode and number of proofs:
`augment(+list(integer),+integer,+compound,-compound)` - `one`

---

augment_arc/4

Pushes the given amount of flow along a single arc, updating the residual dictionary.

Compilation flags:
`static`

Template:
`augment_arc(Id,Amount,Residual0,Residual)`
Mode and number of proofs:
`augment_arc(+integer,+integer,+compound,-compound)` - `one`

---

path_ids/4

Arc identifiers of the path from the BFS source to the given vertex (in source to vertex order).

Compilation flags:

```
static
```

Template:

```
path_ids(Network,Vertex,Info,Ids)
```

Mode and number of proofs:

```
path_ids(+compound,+ground,+compound,-list(integer)) - one
```

---

bottleneck/3

Minimum residual capacity along a non-empty list of arc identifiers.

Compilation flags:

```
static
```

Template:

```
bottleneck(Ids,Residual,Amount)
```

Mode and number of proofs:

```
bottleneck(+list(integer),+compound,-integer) - one
```

---

flow_list/3

Flow on each arc with positive flow as `flow(ArcId, From, To, Amount)` terms in arc input order. `ArcId` is the one-based input arc position.

Compilation flags:

```
static
```

Template:

```
flow_list(Network,Residual,Flow)
```

Mode and number of proofs:

```
flow_list(+compound,+compound,-list(compound)) - one
```

`flow_cost/3`

Total cost of the flow described by the residual dictionary.

Compilation flags:
`static`

Template:
`flow_cost(Network,Residual,Cost)`
Mode and number of proofs:
`flow_cost(+compound,+compound,-integer)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.89.11 `gomory_hu`

Gomory-Hu cut trees using the `avltree` dictionary.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *gomory_hu(avltree)*

Remarks:
(none)

Inherited public predicates:
min_cut_value/4 *tree/2*

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

gomory_hu(Dictionary), *dinic*

object

1.89.12 `gomory_hu(Dictionary)`

- `Dictionary` - Dictionary object used by the flow network.

Gomory-Hu cut trees for networks with symmetric aggregate capacities. Uses n-1 calls to Dinic for n vertices, with no calls for empty networks. Costs and self-loops do not affect cuts.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Uses:
dinic(Dictionary)
flow_network(Dictionary)
list
type

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *tree/2*
  - *min_cut_value/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

tree/2

Returns `tree(Vertices, Edges)` where each `edge(Vertex, Parent, Capacity)` is a weighted tree edge, in vertex order. Requires zero lower bounds and equal total capacity in both directions for every pair of distinct vertices. Parallel multiplicities may differ. Disconnected components are linked by zero-capacity edges.

Compilation flags:

`static`

Template:

`tree(Network,Tree)`

Mode and number of proofs:

`tree(+compound,-compound)` - `one_or_error`

Exceptions:

A lower bound is non-zero:

`domain_error(zero_lower_bounds,Network)`

Aggregate capacities are asymmetric:

`domain_error(symmetric_capacities,Network)`

---

min_cut_value/4

Returns the minimum edge capacity on the unique tree path between distinct vertices. Validates the supplied weighted tree. Adjacency construction and traversal take O(V*log(V)) with balanced dictionaries.

Compilation flags:

`static`

Template:

`min_cut_value(Tree,Source,Sink,Value)`

Mode and number of proofs:

`min_cut_value(+compound,+ground,+ground,-integer)` - `one_or_error`

Exceptions:

An input vertex or the tree is not ground:

```
instantiation_error
```

The tree is malformed:

```
domain_error(gomory_hu_tree,Tree)
```

A terminal is not a tree vertex:

```
existence_error(vertex,Vertex)
```

The terminals are equal:

```
consistency_error(distinct_vertices,Source,Sink)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

gomory_hu, *dinic(Dictionary)*, *flow_network(Dictionary)*

category

1.89.13 `max_flow_common(Dictionary)`

- `Dictionary` - Dictionary object used by the flow network.

Maximum flow and minimum cut predicates shared by the maximum flow algorithm objects. Importing objects must define the `maximize/5` predicate.

Availability:

```
logtalk_load(flow_networks(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:

```
static
```

Implements:

`public` *max_flow_protocol*

Extends:

`public` *flow_support(Dictionary)*

Uses:

flow_network(Dictionary)

list
type
weighted_directed_graph(Dictionary)

Remarks:
(none)

Inherited public predicates:
max_flow/5 *min_cut/5* *min_cuts/5* *multi_source_max_flow/5* *node_capacity_max_flow/6* *sink_closest_min_cut/5* *unique_min_cut/4*

- *Public predicates*
- *Protected predicates*
  - *maximize/5*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

maximize/5

Runs the maximum flow algorithm returning the flow value and the final residual dictionary. Must be defined by the importing objects.

Compilation flags:
`static`

Template:
`maximize(Network,Source,Sink,Value,Residual)`
Mode and number of proofs:
`maximize(+compound,+ground,+ground,-integer,-compound)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.89.14 max_flow_protocol

Maximum flow and minimum cut protocol.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *max_flow/5*
  - *multi_source_max_flow/5*
  - *node_capacity_max_flow/6*
  - *min_cut/5*
  - *sink_closest_min_cut/5*
  - *unique_min_cut/4*
  - *min_cuts/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

max_flow/5

Computes a maximum flow from `Source` to `Sink`. `Flow` is a list of `flow(ArcId, From, To, Amount)` terms with a positive amount, one per arc carrying flow, in the order of the network arcs. `ArcId` is the one-based position of the arc in the input list used to construct the network. Maximum flow currently requires zero lower bounds.

Compilation flags:

`static`

Template:

`max_flow(Network,Source,Sink,Value,Flow)`

Mode and number of proofs:

`max_flow(+compound,+ground,+ground,-integer,-list(compound))` - `one_or_error`

Exceptions:

`Source` or `Sink` is not ground:

`instantiation_error`

`Source` or `Sink` is not a vertex of the network:

`existence_error(vertex,Vertex)`

`Source` and `Sink` are the same vertex:

`consistency_error(distinct_vertices,Source,Sink)`

The network contains a non-zero lower bound:

`domain_error(zero_lower_bounds,Network)`

---

multi_source_max_flow/5

Computes a maximum flow from a non-empty set of distinct sources to a non-empty, disjoint set of distinct sinks. `Flow` uses the original network arc IDs and endpoints. Requires zero lower bounds.

Compilation flags:

`static`

Template:

`multi_source_max_flow(Network,Sources,Sinks,Value,Flow)`

Mode and number of proofs:

`multi_source_max_flow(+compound,+list(ground),+list(ground),-integer,`
`-list(compound))` - `one_or_error`

Exceptions:

A terminal is not ground:

`instantiation_error`

A terminal set is empty:

`domain_error(non_empty_terminal_set,Terminals)`

A terminal is not a vertex of the network:

`existence_error(vertex,Vertex)`

A terminal set contains duplicates:

`domain_error(unique_terminals,Terminals)`

A source is also a sink:

`consistency_error(disjoint_terminals,Sources,Sinks)`

The network contains a non-zero lower bound:

`domain_error(zero_lower_bounds,Network)`

---

node_capacity_max_flow/6

Computes a maximum flow subject to the given intermediate vertex capacities, represented as `Vertex-Capacity` pairs. Source and sink capacities are not constrained. `Flow` uses the original network arc IDs and endpoints. Requires zero lower bounds.

Compilation flags:

`static`

Template:

`node_capacity_max_flow(Network,Source,Sink,NodeCapacities,Value,Flow)`

Mode and number of proofs:

`node_capacity_max_flow(+compound,+ground,+ground,+list(pair),-integer,-list(compound))` - `one_or_error`

Exceptions:

A node capacity vertex is not ground:

`instantiation_error`

A node capacity is not a non-negative integer:

`domain_error(non_negative_integer,Capacity)`

A node capacity vertex is not in the network:

`existence_error(vertex,Vertex)`

A node capacity is specified for a terminal:

`domain_error(intermediate_vertex,Vertex)`

A node capacity is specified more than once for a vertex:

`domain_error(unique_node_capacities,Vertex)`

The network contains a non-zero lower bound:

`domain_error(zero_lower_bounds,Network)`

---

min_cut/5

Computes a minimum cut separating `Source` from `Sink`. `Cut` is a `cut(SourceSide, SinkSide, Arcs)` term where `SourceSide` and `SinkSide` are sorted lists of vertices and `Arcs` is the list of `arc(From, To, Capacity)` terms for the arcs with positive capacity going from the source side to the sink side. `Value` is both the maximum flow value and the capacity of the cut. Minimum cut currently requires zero lower bounds.

Compilation flags:

`static`

Template:

`min_cut(Network,Source,Sink,Value,Cut)`

Mode and number of proofs:

`min_cut(+compound,+ground,+ground,-integer,-compound)` - `one_or_error`

Exceptions:

`Source` or `Sink` is not ground:

`instantiation_error`

`Source` or `Sink` is not a vertex of the network:

`existence_error(vertex,Vertex)`

`Source` and `Sink` are the same vertex:

`consistency_error(distinct_vertices,Source,Sink)`

The network contains a non-zero lower bound:

`domain_error(zero_lower_bounds,Network)`

---

sink_closest_min_cut/5

Computes the sink-closest minimum cut. Its source side contains exactly the vertices that cannot reach the sink in the final residual graph. Requires zero lower bounds.

Compilation flags:

`static`

Template:

`sink_closest_min_cut(Network,Source,Sink,Value,Cut)`

Mode and number of proofs:

`sink_closest_min_cut(+compound,+ground,+ground,-integer,-compound)` - `one_or_error`

Exceptions:

`Source` or `Sink` is not ground:

`instantiation_error`

`Source` or `Sink` is not a vertex of the network:

`existence_error(vertex,Vertex)`

`Source` and `Sink` are the same vertex:

`consistency_error(distinct_vertices,Source,Sink)`

The network contains a non-zero lower bound:

`domain_error(zero_lower_bounds,Network)`

---

unique_min_cut/4

Checks whether the minimum cut separating the given source and sink is unique. `Unique` is `true` or `false`. Requires zero lower bounds.

Compilation flags:

`static`

Template:

`unique_min_cut(Network,Source,Sink,Unique)`

Mode and number of proofs:

`unique_min_cut(+compound,+ground,+ground,-boolean)` - `one_or_error`

Exceptions:

`Source` or `Sink` is not ground:

`instantiation_error`

`Source` or `Sink` is not a vertex of the network:

`existence_error(vertex,Vertex)`

`Source` and `Sink` are the same vertex:

`consistency_error(distinct_vertices,Source,Sink)`

The network contains a non-zero lower bound:

`domain_error(zero_lower_bounds,Network)`

---

min_cuts/5

Enumerates all minimum cuts on backtracking as `cut(SourceSide, SinkSide, Arcs)` terms. Enumeration uses the strongly connected components of the final residual graph. Requires zero lower bounds.

Compilation flags:

`static`

Template:

`min_cuts(Network,Source,Sink,Value,Cut)`

Mode and number of proofs:

`min_cuts(+compound,+ground,+ground,-integer,-compound)` - `zero_or_more_or_error`

Exceptions:

`Source` or `Sink` is not ground:

`instantiation_error`

`Source` or `Sink` is not a vertex of the network:

`existence_error(vertex,Vertex)`

`Source` and `Sink` are the same vertex:

`consistency_error(distinct_vertices,Source,Sink)`

The network contains a non-zero lower bound:

`domain_error(zero_lower_bounds,Network)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

flow_network_protocol, *edmonds_karp(Dictionary)*, *dinic(Dictionary)*, *capacity_scaling(Dictionary)*

protocol

1.89.15 `min_cost_flow_protocol`

Minimum cost flow protocol.

Availability:

```
logtalk_load(flow_networks(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *min_cost_flow/4*
  - *min_cost_flow/5*
  - *min_cost_max_flow/6*
  - *min_cost_max_flow/7*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

min_cost_flow/4

Computes a minimum cost flow satisfying the given supplies and demands, including arc lower bounds. `Supplies` is a list of `Vertex-Amount` pairs where a positive amount is a supply and a negative amount is a demand; vertices not listed are transshipment vertices. Duplicate vertices are added. Fails if there is no feasible flow. Negative arc costs (including negative cost cycles) are supported. `Flow` is a list of `flow(ArcId, From, To, Amount)` terms with a positive amount, one per arc carrying flow, in the order of the network arcs. `ArcId` is the one-based position of the arc in the input list used to construct the network.

Compilation flags:

`static`

Template:

`min_cost_flow(Network,Supplies,Cost,Flow)`

Mode and number of proofs:

`min_cost_flow(+compound,+list(pair),-integer,-list(compound))` - `zero_or_one_or_error`

Exceptions:

`Supplies` is a partial list or contains a variable vertex:

`instantiation_error`

`Supplies` is not a list of `Vertex-Amount` pairs:

`type_error(list(pair),Supplies)`

An amount is not an integer:

`type_error(integer,Amount)`

A vertex of `Supplies` is not a vertex of the network:

`existence_error(vertex,Vertex)`

The supplies and demands do not add up to zero:

`domain_error(balanced_supplies,Supplies)`

---

min_cost_flow/5

As `min_cost_flow/4`, also returning the final vertex potentials as `Vertex-Potential` pairs in vertex order. The potentials certify non-negative reduced costs on the residual arcs.

Compilation flags:

`static`

Template:

`min_cost_flow(Network,Supplies,Cost,Flow,Potentials)`

Mode and number of proofs:

`min_cost_flow(+compound,+list(pair),-integer,-list(compound),-list(pair))` - `zero_or_one_or_error`

Exceptions:

`Supplies` is a partial list or contains a variable vertex:

`instantiation_error`

`Supplies` is not a list of `Vertex-Amount` pairs:

`type_error(list(pair),Supplies)`

An amount is not an integer:

`type_error(integer,Amount)`

A vertex of `Supplies` is not a vertex of the network:

`existence_error(vertex,Vertex)`

The supplies and demands do not add up to zero:

`domain_error(balanced_supplies,Supplies)`

---

min_cost_max_flow/6

Computes a minimum cost flow among the maximum flows from `Source` to `Sink`. Always succeeds for valid terminals. Currently requires zero lower bounds.

Compilation flags:

`static`

Template:

`min_cost_max_flow(Network,Source,Sink,Value,Cost,Flow)`

Mode and number of proofs:

`min_cost_max_flow(+compound,+ground,+ground,-integer,-integer,-list(compound))` - `one_or_error`

Exceptions:

`Source` or `Sink` is not ground:

`instantiation_error`

`Source` or `Sink` is not a vertex of the network:

`existence_error(vertex,Vertex)`

`Source` and `Sink` are the same vertex:

`consistency_error(distinct_vertices,Source,Sink)`

The network contains a non-zero lower bound:

`domain_error(zero_lower_bounds,Network)`

---

min_cost_max_flow/7

As `min_cost_max_flow/6`, also returning final vertex potentials as `Vertex-Potential` pairs in vertex order.

Compilation flags:

`static`

Template:

`min_cost_max_flow(Network,Source,Sink,Value,Cost,Flow,Potentials)`

Mode and number of proofs:

`min_cost_max_flow(+compound,+ground,+ground,-integer,-integer,-list(compound),-list(pair))` - `one_or_error`

Exceptions:

`Source` or `Sink` is not ground:

`instantiation_error`

`Source` or `Sink` is not a vertex of the network:

`existence_error(vertex,Vertex)`

`Source` and `Sink` are the same vertex:

`consistency_error(distinct_vertices,Source,Sink)`

The network contains a non-zero lower bound:

`domain_error(zero_lower_bounds,Network)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

flow_network_protocol, *max_flow_protocol*, *successive_shortest_paths(Dictionary)*

object

1.89.16 `piecewise_costs`

Convex piecewise-linear integer cost network builder using the `avltree` dictionary.

Availability:

`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:

`static, context_switching_calls`

Extends:

`public` *piecewise_costs(avltree)*

Remarks:

(none)

Inherited public predicates:

build/4 *collapse_flow/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

piecewise_costs(Dictionary), *successive_shortest_paths*

object

1.89.17 `piecewise_costs(Dictionary)`

- `Dictionary` - Dictionary object used by the expanded flow network (e.g., `avltree`).

Builds ordinary flow networks that model convex piecewise-linear integer arc costs by expanding each cost segment into a linear-cost arc. The returned mapping is opaque and records the public vertices and arcs of the expanded network.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Uses:
flow_network(Dictionary)
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *build/4*
  - *collapse_flow/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

build/4

Expands convex cost arcs `arc(From, To, Lower, Segments)` where each segment is `segment(Width, Marginal-Cost)`. Widths are positive integers and marginal costs are nondecreasing integers; the original cost at flow amount `X` is the sum of the marginal costs of its first `X` units, with cost zero at amount zero. Each expanded arc receives the lower-bound portion of the original flow in segment order. The opaque mapping records a ground public network snapshot (vertices and arcs), and preserves original arc and segment order.

Compilation flags:
`static`

Template:
`build(Vertices,CostArcs,Network,Mapping)`

Mode and number of proofs:
`build(+list(ground),+list(compound),-compound,-compound)` - `one_or_error`

Exceptions:
An input list is partial, or a vertex is not ground:
`instantiation_error`
An input is not a list:
`type_error(list,Input)`
A cost arc is not an `arc/4` term:
`domain_error(piecewise_cost_arc,Arc)`
A segment is not a `segment/2` term:
`domain_error(piecewise_cost_segment,Segment)`
A lower bound is not an integer:
`type_error(integer,Lower)`
A lower bound is negative:
`domain_error(non_negative_integer,Lower)`
A lower bound exceeds the total segment width:
`consistency_error(piecewise_lower_bound_and_capacity,Lower,Capacity)`
A segment width is not an integer:
`type_error(integer,Width)`

A segment width is not positive:
`domain_error(positive_integer,Width)`
A marginal cost is not an integer:
`type_error(integer,MarginalCost)`
Segment marginal costs decrease:
`domain_error(non_decreasing_marginal_costs,Segments)`
A cost arc has no segments:
`domain_error(non_empty_list,Segments)`

---

collapse_flow/4

Projects a capacity-valid flow on the expanded network to original `flow(ArcId, From, To, Amount)` entries, omitting zero flows. Rejects a noncanonical segment allocation when a more expensive segment carries optional flow while a cheaper segment has unused capacity. Arbitrary distribution across equal-cost segments is accepted. The original convex arcs are not themselves a core flow network.

Compilation flags:
`static`

Template:
`collapse_flow(Network,Mapping,ExpandedFlow,Flow)`
Mode and number of proofs:
`collapse_flow(+compound,+compound,+list(compound),-list(compound))` - `one_or_error`

Exceptions:
The mapping is not ground:
`instantiation_error`
The mapping is malformed:
`domain_error(piecewise_cost_mapping,Mapping)`
The network vertices or arcs differ from the mapped snapshot:
`consistency_error(piecewise_mapping_network,Snapshot,MappedSnapshot)`
The expanded flow is a variable or partial list, or an entry has a variable field:
`instantiation_error`
The expanded flow is not a list:
`type_error(list,ExpandedFlow)`
An expanded-flow entry is malformed:
`domain_error(flow,Term)`
An expanded arc ID or flow amount is not an integer:
`type_error(integer,Value)`
An expanded arc ID or flow amount is not positive:
`domain_error(positive_integer,Value)`
An expanded arc ID is not in the network:
`existence_error(arc,ArcId)`
An expanded-flow entry has incorrect endpoints:
`domain_error(flow_arc_endpoints,Term)`
The expanded flow violates an arc bound or repeats an arc ID:
`domain_error(capacity_valid_flow,ExpandedFlow)`

The segment allocation is not canonical:
`domain_error(canonical_piecewise_flow,ExpandedFlow)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

flow_network(Dictionary), *successive_shortest_paths(Dictionary)*

object

1.89.18 `successive_shortest_paths`

Minimum cost flow using the successive shortest paths algorithm and the `avltree` dictionary.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *successive_shortest_paths(avltree)*

Remarks:
(none)

Inherited public predicates:
min_cost_flow/4 *min_cost_flow/5* *min_cost_max_flow/6* *min_cost_max_flow/7*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

successive_shortest_paths(Dictionary), *dinic*

object

1.89.19 `successive_shortest_paths(Dictionary)`

- `Dictionary` - Dictionary object used by the flow network (e.g., `avltree`).

Minimum cost flow using the successive shortest paths algorithm with node potentials (reduced costs), where each shortest path is found using Dijkstra algorithm and a binary heap. Arcs with negative cost are saturated beforehand so that all residual costs are non-negative, which also handles negative cost cycles. Supplies and demands are routed from a super source to a super sink. Integer capacities and costs only.

Availability:
`logtalk_load(flow_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *min_cost_flow_protocol*
Imports:
`public` *flow_support(Dictionary)*
Uses:
binary_heap_min
dinic(Dictionary)
flow_network(Dictionary)
type

Remarks:
(none)

Inherited public predicates:

min_cost_flow/4 *min_cost_flow/5* *min_cost_max_flow/6* *min_cost_max_flow/7*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dinic(Dictionary), *flow_network(Dictionary)*, *successive_shortest_paths*

object

1.89.20 `time_expanded_network`

Builds time-expanded flow networks using the `avltree` dictionary.

Availability:

```
logtalk_load(flow_networks(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *time_expanded_network(avltree)*

Remarks:

(none)

Inherited public predicates:

build/6

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

time_expanded_network(Dictionary)

object

1.89.21 `time_expanded_network(Dictionary)`

- `Dictionary` - Dictionary object used by the constructed flow network.

Builds finite, discrete-time flow networks from explicit departure and storage schedules. Expanded vertices are tagged `time(Vertex, Time)` terms; schedule arcs retain input order and arc IDs.

Availability:

```
logtalk_load(flow_networks(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-16

Compilation flags:

```
static, context_switching_calls
```

Uses:

flow_network(Dictionary)
list

type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *build/6*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

build/6

Builds a time-expanded flow network. `Vertices` is a proper list of ground base vertices and `Horizon` is a non-negative integer. Each `travel(From, To, Departure, Duration, Lower, Capacity, Cost)` creates one arc from `time(From,Departure)` to `time(To,Departure+Duration)`. Each `store(Vertex, Time, Lower, Capacity, Cost)` creates one arc from `time(Vertex,Time)` to `time(Vertex,Time+1)`. Durations are positive; schedule endpoints and arrivals must be within the horizon. Missing storage entries mean no waiting capacity. The mapping is `mapping(TravelEntries, StorageEntries)` where entries are `travel(ArcId, EntryId, From, To, Departure, Arrival)` and `store(ArcId, EntryId, Vertex, Time, Arrival)`; entry IDs are one-based in their respective input lists and arc IDs follow travel input order then storage input order. Expanded vertices are tagged `time(Vertex, Time)` terms, preserving all time copies including isolated vertices. Supplies and demands at exact times can be passed to the existing minimum-cost flow solver. Evacuation by a deadline can be modeled by adding collection arcs from eligible time copies to a fresh terminal. Storage capacities constrain individual intervals, not a shared per-vertex capacity. Non-zero lower bounds require the supply-based minimum-cost flow solver; maximum-flow APIs require zero lower bounds.

Compilation flags:
`static`

Template:
`build(Vertices,Horizon,Travels,Storage,Network,Mapping)`

Mode and number of proofs:
`build(+list(ground),+integer,+list(compound),+list(compound),--compound,--compound)` - `one_or_error`

Exceptions:
`Vertices`, `Travels`, or `Storage` is a variable or partial list, or a base vertex or schedule endpoint is non-ground:
`instantiation_error`
An input list is neither a variable nor a list:
`type_error(list,Input)`
`Horizon`, a departure, a duration, a storage time, a lower bound, a capacity, or a cost is not an integer:
`type_error(integer,Value)`
`Horizon` is negative:
`domain_error(non_negative_integer,Horizon)`

A travel or storage entry has an invalid shape:

```
domain_error(time_expanded_schedule,Entry)
```

A scheduled endpoint is not in `Vertices`:

```
existence_error(vertex,Vertex)
```

A departure or storage time is outside the horizon, or a travel duration exceeds the remaining horizon:

```
consistency_error(time_expanded_schedule_and_horizon,Entry,Horizon)
```

A travel duration is not positive:

```
domain_error(positive_integer,Duration)
```

A lower bound or capacity is negative:

```
domain_error(non_negative_integer,Value)
```

A lower bound exceeds its capacity:

```
consistency_error(arc_lower_bound_and_capacity,Lower,Capacity)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

time_expanded_network, *flow_network(Dictionary)*, *successive_shortest_paths(Dictionary)*, *dinic(Dictionary)*

1.90 format

object

1.90.1 `format`

Formatted output predicates.

Availability:

```
logtalk_load(format(loader))
```

Author: Paulo Moura
Version: 1:4:0
Date: 2026-10-28

Compilation flags:

```
static, context_switching_calls
```

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *format/3*
  - *format/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

format/3

Writes a list of arguments after a format specification to the specified output stream.

Compilation flags:
`static`

Template:
`format(Stream,Format,Arguments)`

Mode and number of proofs:
`format(@stream_or_alias,+atom,@list)` - `zero_or_one`
`format(@stream_or_alias,+list(character_code),@list)` - `zero_or_one`
`format(@stream_or_alias,+list(character),@list)` - `zero_or_one`

---

format/2

Writes a list of arguments after a format specification to the current output stream.

Compilation flags:
`static`

Template:
`format(Format,Arguments)`

Mode and number of proofs:
`format(+atom,@list)` - `zero_or_one`
`format(+list(character_code),@list)` - `zero_or_one`
`format(+list(character),@list)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.91 fp_growth_pattern_miner

object

1.91.1 `fp_growth_pattern_miner`

FP-growth frequent itemset miner for transaction datasets using recursive conditional pattern-base projection over an FP-tree with header-table and parent-link navigation.

Availability:
`logtalk_load(fp_growth_pattern_miner(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-05-20

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *frequent_pattern_mining_common*
Uses:
format
list

Remarks:
(none)

Inherited public predicates:
check_option/1 check_options/1 check_pattern_miner/1 default_option/1 default_options/1 diagnostic/2 diagnostics/2 export_to_clauses/4 export_to_file/4 mine/2 mine/3 option/2 option/3 pattern_miner_options/2 print_pattern_miner/1 valid_option/1 valid_options/1 valid_pattern_miner/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pattern_miner_protocol, *transaction_dataset_protocol*, *apriori_pattern_miner*

1.92 frequent_pattern_mining_protocols

category

1.92.1 `frequent_pattern_mining_common`

Shared predicates for frequent itemset miner dataset validation, support accumulation, and itemset ordering/filtering helpers.

Availability:

```
logtalk_load(frequent_pattern_mining_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-02

Compilation flags:

```
static
```

Extends:

```
public
```
pattern_miner_common

Uses:

list

type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_pattern_miner/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *mine/2* *mine/3* *option/2* *option/3* *pattern_miner_options/2* *print_pattern_miner/1* *valid_option/1* *valid_options/1* *valid_pattern_miner/1*

- *Public predicates*
- *Protected predicates*
  - *check_transactions/4*
  - *sort_patterns/2*
  - *filter_patterns/3*
  - *count_items/4*
  - *select_frequent_item_supports/3*
  - *valid_itemset_patterns/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

check_transactions/4

Validates the collected dataset transactions and returns the maximum transaction length.

Compilation flags:

```
static
```

Template:

```
check_transactions(Dataset,ItemDomain,Transactions,MaxTransactionLength)
```

Mode and number of proofs:

```
check_transactions(+object_identifier,+list(atom),+list(pair(integer,
list(atom))),-integer) - one
```

---

sort_patterns/2

Sorts itemset patterns first by pattern length and then lexicographically.

Compilation flags:

```
static
```

Template:

```
sort_patterns(Patterns0,Patterns)
```

Mode and number of proofs:

```
sort_patterns(+list(compound),-list(compound)) - one
```

---

filter_patterns/3

Filters itemset patterns by minimum pattern length.

Compilation flags:
```
static
```

Template:
```
filter_patterns(Patterns0,MinimumPatternLength,Patterns)
```
Mode and number of proofs:
```
filter_patterns(+list(compound),+integer,-list(compound)) - one
```

---

count_items/4

Accumulates support counts for a list of items.

Compilation flags:
```
static
```

Template:
```
count_items(Items,Count,ItemCounts0,ItemCounts)
```
Mode and number of proofs:
```
count_items(+list(atom),+integer,+list(pair(atom,integer)),-list(pair(atom,
integer))) - one
```

---

select_frequent_item_supports/3

Selects the item supports that satisfy the minimum support count.

Compilation flags:
```
static
```

Template:
```
select_frequent_item_supports(ItemCounts,SupportCount,FrequentItemSupports)
```
Mode and number of proofs:
```
select_frequent_item_supports(+list(pair(atom,integer)),+integer,-list(compound))
- one
```

`valid_itemset_patterns/2`

True when the patterns are valid `itemset(Items, Support)` terms over the given item domain with positive integer supports.

Compilation flags:
`static`

Template:
`valid_itemset_patterns(ItemDomain,Patterns)`
Mode and number of proofs:
`valid_itemset_patterns(+list(atom),+list(compound))` - `zero_or_one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.92.2 `transaction_dataset_protocol`

Protocol for transaction datasets used with frequent itemset mining algorithms. Transaction identifiers are expected to be unique within a dataset.

Availability:
`logtalk_load(frequent_pattern_mining_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-29

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

    - *items/1*
    - *transaction/2*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

items/1

Returns the declared transaction item domain as a canonical sorted list of unique items.

Compilation flags:
```
static
```

Template:
```
items(Items)
```
Mode and number of proofs:
```
items(-list(atom)) - one
```

---

transaction/2

Enumerates by backtracking the transactions in the dataset. The transaction identifier is expected to be unique within the dataset. Each transaction is represented as a canonical sorted list of unique declared items.

Compilation flags:
```
static
```

Template:
```
transaction(Id,Transaction)
```
Mode and number of proofs:
```
transaction(-integer,-list(atom)) - zero_or_more
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.93 gaussian_mixture_clusterer

object

1.93.1 `gaussian_mixture_clusterer`

Gaussian mixture model clusterer for continuous datasets. Learns from a dataset object implementing the `clustering_dataset_protocol` protocol and returns a clusterer term that can be used for assigning new instances to clusters and exported as predicate clauses.

Availability:
`logtalk_load(gaussian_mixture_clusterer(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *clusterer_common*
Uses:
format
list
numberlist
population
type

Remarks:
(none)

Inherited public predicates:
check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *cluster_probabilities/3*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

`cluster_probabilities/3`

Returns posterior component probabilities for a new instance as `Cluster-Probability` pairs in component-id order.

Compilation flags:
`static`

Template:
`cluster_probabilities(Clusterer,Instance,Probabilities)`
Mode and number of proofs:
`cluster_probabilities(+compound,+list,-list(pair))` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *kmeans_clusterer*

1.94 gaussian_process_regression

object

1.94.1 `gaussian_process_regression`

Gaussian process regression regressor supporting continuous and mixed-feature datasets using an exact mixed Gaussian process with posterior uncertainty estimates. Learns from a dataset object implementing the `regression_dataset_protocol` protocol and returns a regressor term that can be used for prediction, predictive-distribution queries, and export as predicate clauses.

Availability:
`logtalk_load(gaussian_process_regression(loader))`

Author: Paulo Moura

Version: 1:1:2
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *regressor_common*

Uses:
format
linear_algebra
list
numberlist
pairs
population
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
  - *predict_distribution/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

predict_distribution/3

Predicts the posterior predictive Gaussian distribution for a new instance using the learned regressor. The returned term has the shape `gaussian(Mean, Variance)` where `Variance` includes the learned observation noise variance.

Compilation flags:
`static`

Template:
`predict_distribution(Regressor,Instance,Distribution)`
Mode and number of proofs:
`predict_distribution(+compound,+list,-compound)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

linear_regression, *ridge_regression*, *lasso_regression*, *elastic_net_regression*, *regression_tree*, *random_forest_regression*

1.95 genetic_algorithm

object

1.95.1 `genetic_algorithm(Problem)`

- `Problem` - Problem object implementing `genetic_algorithm_problem_protocol`.

Genetic algorithm meta-heuristic optimizer using the Xoshiro128++ random number generator via `fast_random(xoshiro128pp)`.

Availability:

```
logtalk_load(genetic_algorithm(loader))
```

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *genetic_algorithm(Problem,fast_random(xoshiro128pp))*

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

genetic_algorithm(Problem,Random), *genetic_algorithm_problem_protocol*

object

1.95.2 `genetic_algorithm(Problem,Random)`

- `Problem` - Problem object implementing `genetic_algorithm_problem_protocol`.
- `Random` - Random number generator object providing `between/3`, `random/1`, and `randomize/1` predicates.

Genetic algorithm meta-heuristic optimizer. Parameterized by a problem object implementing the `genetic_algorithm_problem_protocol` protocol and by a random number generator algorithm for the `fast_random` library. The algorithm minimizes the energy (cost) function defined by the problem by default; maximization is supported via options. Custom stop conditions, diversity measures, progress reporting, and selection pressure can be defined by the problem object or configured via options; suitable defaults are used otherwise.

Availability:

```
logtalk_load(genetic_algorithm(loader))
```

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public options
```

Uses:

list
numberlist
type

Remarks:

- Algorithm: A generational genetic algorithm that maintains a population of individuals. Each generation applies selection, crossover, and mutation to produce the next population. Optional elitism preserves the best individuals across generations.

- Selection: Parent selection is controlled by the `selection/1` option. Supported schemes are `tournament(K)` (default `tournament(3)`), `roulette`, and `rank`. Tournament samples K individuals and keeps the best; roulette selects proportionally to fitness derived from energy; rank selects proportionally to rank after sorting by objective.
- Crossover and mutation: Both operators are defined by the problem object. Crossover is applied with probability `crossover_rate`; mutation is applied independently to each offspring with probability `mutation_rate`. Rates may be held constant or adapted each generation via `crossover_schedule/1` and `mutation_schedule/1` options, or via optional problem hooks `crossover_rate/4` and `mutation_rate/4` (hooks take precedence over schedules).
- Elitism: When `elite_size(N)` is greater than zero, the best N individuals of the current population (clamped to the population size) are copied unchanged into the next generation. This preserves the best solutions found so far against disruption by crossover and mutation. Set `elite_size(0)` to disable elitism.
- Best individual tracking: The algorithm tracks the best individual found across all generations, not just the final population.
- Seed control: The `seed(S)` option initializes the random number generator for reproducible runs.
- Progress reporting: If the problem object defines `progress/5`, it is called periodically with the current generation, best individual, best energy, mean population energy, and diversity. A final report is always produced when the loop terminates when updates are enabled.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *run/2*
  - *run/3*
  - *run/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

run/2

Runs the genetic algorithm using default options and returns the best individual found and its energy.

Compilation flags:

`static`

Template:

`run(BestIndividual,BestEnergy)`

Mode and number of proofs:

`run(-nonvar,-number)` - `one`

run/3

Runs the genetic algorithm using the given options and returns the best individual found and its energy.

Compilation flags:
`static`

Template:
`run(BestIndividual,BestEnergy,Options)`
Mode and number of proofs:
`run(-nonvar,-number,+list(compound))` - `one`

Remarks:

- `max_generations(N)` option: Maximum number of generations (default: `200`).
- `population_size(N)` option: Number of individuals in the population; must be at least 2 (default: `50`).
- `crossover_rate(P)` option: Initial probability of applying crossover to a selected pair of parents (default: `0.8`). Used as the constant rate when the schedule is `constant`, and as the starting value for adaptive schedules.
- `mutation_rate(P)` option: Initial probability of mutating each offspring (default: `0.1`). Used as the constant rate when the schedule is `constant`, and as the starting value for adaptive schedules.
- `crossover_schedule(Schedule)` option: How the crossover rate evolves across generations: `constant` (default), `linear(Initial, Final)` interpolates from `Initial` to `Final`, or `geometric(Factor)` multiplies the rate by `Factor` each generation (clamped to `[0.0, 1.0]`). Overridden when the problem defines `crossover_rate/4`.
- `mutation_schedule(Schedule)` option: How the mutation rate evolves across generations: `constant` (default), `linear(Initial, Final)` interpolates from `Initial` to `Final`, or `geometric(Factor)` multiplies the rate by `Factor` each generation (clamped to `[0.0, 1.0]`). Overridden when the problem defines `mutation_rate/4`.
- `selection(Scheme)` option: Parent selection scheme: `tournament(K)` with positive integer K (default: `tournament(3)`), `roulette`, or `rank`.
- `elite_size(N)` option: Number of best individuals preserved unchanged into the next generation (default: `1`). Set to `0` to disable elitism.
- `objective(Direction)` option: Optimization direction: `minimize` (default) or `maximize`.
- `updates(N)` option: Number of progress reports during the run. Set to `0` to disable. Progress is reported by calling `progress/5` on the problem object (default: `0`).
- `seed(S)` option: Positive integer seed for the random number generator, enabling reproducible runs (default: none).

---

run/4

Runs the genetic algorithm using the given options, returns the best individual found and its energy, and returns run statistics.

Compilation flags:
`static`

Template:
`run(BestIndividual,BestEnergy,Statistics,Options)`
Mode and number of proofs:
`run(-nonvar,-number,-list(compound),+list(compound))` - `one`

Remarks:

- Statistics list: A list of `Key(Value)` pairs: `generations(N)` is the number of generations executed, `evaluations(E)` is the total number of fitness evaluations, `improvements(I)` is the number of generations that improved the best energy, and `final_population_size(S)` is the size of the final population.

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

genetic_algorithm(Problem), *genetic_algorithm_problem_protocol*

protocol

1.95.3 `genetic_algorithm_problem_protocol`

Protocol for genetic algorithm problem definitions. A problem object must define the required predicates for individuals, fitness (energy), crossover, and mutation, and may optionally define initial population generation, stopping, progress reporting, diversity, and adaptive rate hooks.

Availability:
`logtalk_load(genetic_algorithm(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-16

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

      - *random_individual/1*
      - *initial_population/1*
      - *state_energy/2*
      - *crossover/4*
      - *mutate/2*
      - *stop_condition/3*
      - *progress/5*
      - *diversity/2*
      - *crossover_rate/4*
      - *mutation_rate/4*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

random_individual/1

Generates a random individual (candidate solution). Used to build the initial population when `initial_population/1` is not defined and to replace individuals when needed.

Compilation flags:
`static`

Template:
`random_individual(Individual)`
Mode and number of proofs:
`random_individual(-nonvar)` - `one`

---

initial_population/1

Optionally returns a non-empty list of initial individuals. When not defined, the algorithm generates a random initial population of the requested size by repeated calls to `random_individual/1`.

Compilation flags:
`static`

Template:
`initial_population(Population)`
Mode and number of proofs:
`initial_population(-list(nonvar))` - `zero_or_one`

---

state_energy/2

Computes the energy (cost / fitness) of the given individual. The algorithm minimizes this value by default; use the `objective(maximize)` option to maximize it instead.

Compilation flags:
`static`

Template:
`state_energy(Individual,Energy)`
Mode and number of proofs:
`state_energy(+nonvar,-number)` - `one`

---

crossover/4

Recombines two parent individuals into two offspring. This is the most problem-specific operator after representation and strongly influences search quality.

Compilation flags:
`static`

Template:
`crossover(Parent1,Parent2,Offspring1,Offspring2)`
Mode and number of proofs:
`crossover(+nonvar,+nonvar,-nonvar,-nonvar)` - `one`

---

mutate/2

Produces a mutated version of the given individual. Mutation introduces diversity and helps escape local optima.

Compilation flags:
`static`

Template:
`mutate(Individual,Mutated)`
Mode and number of proofs:
`mutate(+nonvar,-nonvar)` - `one`

---

stop_condition/3

True when the search should stop given the current generation, best individual found so far, and its energy. Optional. When not defined by the problem, the search runs until the maximum number of generations is reached.

Compilation flags:

```
static
```

Template:

```
stop_condition(Generation,BestIndividual,BestEnergy)
```

Mode and number of proofs:

```
stop_condition(+non_negative_integer,+nonvar,+number) - zero_or_one
```

---

progress/5

Called periodically to report optimization progress. Optional. When not defined by the problem, progress reporting is skipped. The mean energy and diversity are population statistics; diversity is problem-defined when `diversity/2` is present, otherwise a simple placeholder.

Compilation flags:

```
static
```

Template:

```
progress(Generation,BestIndividual,BestEnergy,MeanEnergy,Diversity)
```

Mode and number of proofs:

```
progress(+non_negative_integer,+nonvar,+number,+number,+number) - zero_or_one
```

---

diversity/2

Optionally computes a numeric diversity measure for the current population. When not defined, progress reporting uses 0.0 for diversity.

Compilation flags:

```
static
```

Template:

```
diversity(Population,Diversity)
```

Mode and number of proofs:

```
diversity(+list(nonvar),-number) - zero_or_one
```

---

crossover_rate/4

Optional adaptive crossover-rate hook. Called once per generation with the current generation index, the maximum number of generations, and the current crossover rate. When defined and successful, the returned rate (clamped to `[0.0, 1.0]`) is used for that generation and overrides any `crossover_schedule/1` option. When not defined or when it fails, the configured schedule is applied instead.

Compilation flags:

```
static
```

Template:

```
crossover_rate(Generation,MaxGenerations,CurrentRate,NewRate)
```

Mode and number of proofs:

```
crossover_rate(+non_negative_integer,+positive_integer,+float,-float) -
zero_or_one
```

---

mutation_rate/4

Optional adaptive mutation-rate hook. Called once per generation with the current generation index, the maximum number of generations, and the current mutation rate. When defined and successful, the returned rate (clamped to `[0.0, 1.0]`) is used for that generation and overrides any `mutation_schedule/1` option. When not defined or when it fails, the configured schedule is applied instead.

Compilation flags:

```
static
```

Template:

```
mutation_rate(Generation,MaxGenerations,CurrentRate,NewRate)
```

Mode and number of proofs:

```
mutation_rate(+non_negative_integer,+positive_integer,+float,-float) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

genetic_algorithm(Problem), *genetic_algorithm(Problem,Random)*

1.96 genint

object

1.96.1 genint

Global object for generating increasing non-negative integers for named counters. The predicates are declared as synchronized when the library is compiled using a backend supporting threads.

Availability:
`logtalk_load(genint(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2022-07-21

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *genint_core*

Remarks:
(none)

Inherited public predicates:
genint/2 *reset_genint/0* *reset_genint/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.96.2 genint_core

Predicates for generating increasing non-negative integers. The predicates are declared as synchronized when the library is compiled using a backend supporting threads.

Availability:
`logtalk_load(genint(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2022-07-26

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *reset_genint/0*
  - *reset_genint/1*
  - *genint/2*
- *Protected predicates*
- *Private predicates*
  - *counter_/2*
- *Operators*

Public predicates

reset_genint/0

Resets all counters.

Compilation flags:
```
static, synchronized
```

Mode and number of proofs:
```
reset_genint - one
```

---

reset_genint/1

Resets the given counter.

Compilation flags:
```
static, synchronized
```

Template:
```
reset_genint(Counter)
```
Mode and number of proofs:
```
reset_genint(+atom) - one
```

---

genint/2

Returns the next integer for a given counter.

Compilation flags:
```
static, synchronized
```

Template:
```
genint(Counter,Integer)
```
Mode and number of proofs:
```
genint(+atom,-non_negative_integer) - one
```

Protected predicates

(none)

Private predicates

counter_/2

Table of current state of counters.

Compilation flags:
`dynamic`

Template:
`counter_(Counter,Latest)`
Mode and number of proofs:
`counter_(?atom,?non_negative_integer)` - `zero_or_more`

Operators

(none)

1.97 gensym

object

1.97.1 gensym

Global object for generating unique atoms. The predicates are declared as synchronized when the library is compiled using a backend supporting threads.

Availability:
`logtalk_load(gensym(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2022-07-21

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *gensym_core*

Remarks:
(none)

Inherited public predicates:

gensym/2 *reset_gensym/0* *reset_gensym/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

category

1.97.2 `gensym_core`

Predicates for generating unique atoms. Protocol based on the `gensym` module of SWI-Prolog. The predicates are declared as synchronized when the library is compiled using a backend supporting threads.

Availability:

```
logtalk_load(gensym(loader))
```

Author: Paulo Moura
Version: 2:1:0
Date: 2022-07-26

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *reset_gensym/0*
  - *reset_gensym/1*
  - *gensym/2*
- *Protected predicates*
- *Private predicates*
  - *base_/2*
- *Operators*

Public predicates

reset_gensym/0

Resets the generator counter for all bases.

Compilation flags:
```
static, synchronized
```

Mode and number of proofs:
```
reset_gensym - one
```

---

reset_gensym/1

Resets the generator counter for a given base.

Compilation flags:
```
static, synchronized
```

Template:
```
reset_gensym(Base)
```
Mode and number of proofs:
```
reset_gensym(+atom) - one
```

---

gensym/2

Returns a new unique atom with a given base (prefix).

Compilation flags:
```
static, synchronized
```

Template:
```
gensym(Base,Unique)
```

Mode and number of proofs:
`gensym(+atom,-atom)` - `one`

Protected predicates

(none)

Private predicates

`base_/2`

Table of generator bases and respective counters.

Compilation flags:
`dynamic`

Template:
`base_(Base,Counter)`
Mode and number of proofs:
`base_(?atom,?integer)` - `zero_or_more`

Operators

(none)

1.98 geohash

object

1.98.1 `geohash`

Geohash encoder, decoder, adjacency, covering, hierarchy, integer conversion, bit-precision, and geometry covering predicates for `geographic(Latitude,Longitude)` coordinates.

Availability:
`logtalk_load(geohash(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2026-10-28

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *geohash_protocol*

Imports:

`public` *options*

Uses:

geospatial
list

Remarks:

- Alphabet: Uses the standard geohash base-32 alphabet `0123456789bcdefghjkmnpqrstuvwxyz`.
- Encoding canonicalization: Encoding canonicalizes longitudes to the `[-180.0,180.0[` range, treating `180.0` as `-180.0` so equivalent dateline coordinates map to the same geohash.
- Bounding boxes: Decoded cells are represented using `bbox(geographic(MinLatitude,MinLongitude),geographic(MaxLatitude,MaxLongitude))`.
- Coverage: Bounding-box coverage accepts antimeridian-crossing boxes where the minimum longitude is greater than the maximum longitude.
- Adaptive covering: Adaptive covers use `max_precision/1` together with `compact/1` and `min_precision/1` options.
- Integer representation: Integer geohashes are 5-bit packed base-32 values with an explicit precision argument preserving leading zero groups.
- Geometry covering: Polygon and polyline covering predicates rely on generic `geospatial` geometry predicates and do not support antimeridian-crossing input geometries.

Inherited public predicates:

adjacent/3 adjacent_bits/4 adjacent_int/4 bounding_box/2 bounding_box_bits/3 bounding_box_int/3 cell_dimensions/3 check_option/1 check_options/1 children/2 common_prefix/3 compress/2 covering/3 covering/4 decode/2 decode_bits/3 decode_int/3 default_option/1 default_options/1 encode/3 encode_bits/3 encode_int/3 expand/2 expand_int/3 geohash_to_int/2 int_to_geohash/3 neighbors/2 neighbors_bits/3 neighbors_int/3 option/2 option/3 parent/2 polygon_covering/4 polyline_covering/4 precision/3 valid_geohash/1 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

geohash_protocol, *geospatial*

protocol

1.98.2 geohash_protocol

Geohash predicates protocol.

Availability:

```
logtalk_load(geohash(loader))
```

Author: Paulo Moura
Version: 1:3:0
Date: 2026-10-11

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

- Spatial precision: Spatial predicates accept 1 to 12 characters or 1 to 60 bits; greater precision fails to avoid collapsed floating-point cell bounds.
- Integer capacity: Packed integer predicates additionally require the exclusive upper bound to fit the backend integer range.
- Hierarchy precision: Text validation, parent, children, common prefix, and compression accept longer hashes.

Inherited public predicates:

(none)

- *Public predicates*
  - *valid_geohash/1*
  - *encode/3*
  - *decode/2*
  - *bounding_box/2*
  - *precision/3*
  - *cell_dimensions/3*
  - *adjacent/3*

- neighbors/2
- covering/3
- covering/4
- compress/2
- parent/2
- children/2
- common_prefix/3
- encode_int/3
- decode_int/3
- bounding_box_int/3
- adjacent_int/4
- neighbors_int/3
- geohash_to_int/2
- int_to_geohash/3
- encode_bits/3
- decode_bits/3
- bounding_box_bits/3
- adjacent_bits/4
- neighbors_bits/3
- expand/2
- expand_int/3
- polygon_covering/4
- polyline_covering/4

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

valid_geohash/1

True when the argument is a non-empty geohash atom using the standard geohash base-32 alphabet `0123456789bcdefghjkmnpqrstuvwxyz`.

Compilation flags:

`static`

Template:

`valid_geohash(Geohash)`

Mode and number of proofs:

`valid_geohash(+atom)` - `zero_or_one`

encode/3

Encodes a geographic coordinate represented as `geographic(Latitude,Longitude)` into a geohash atom with the given precision in characters. Longitude is canonicalized to the `[-180.0,180.0[` range, so `180.0` is treated as `-180.0`.

Compilation flags:
`static`

Template:
`encode(Coordinate,Precision,Geohash)`
Mode and number of proofs:
`encode(+compound,+positive_integer,-atom)` - `zero_or_one`

---

decode/2

Decodes a geohash atom into the center geographic coordinate of the represented cell.

Compilation flags:
`static`

Template:
`decode(Geohash,Coordinate)`
Mode and number of proofs:
`decode(+atom,-compound)` - `zero_or_one`

---

bounding_box/2

Decodes a geohash atom into its geographic bounding box represented as `bbox(geographic(MinLatitude,MinLongitude),geographic(MaxLatitude,MaxLongitude))`.

Compilation flags:
`static`

Template:
`bounding_box(Geohash,BoundingBox)`
Mode and number of proofs:
`bounding_box(+atom,-compound)` - `zero_or_one`

---

precision/3

Returns the maximum latitude and longitude errors in degrees for geohashes of the given precision in characters.

Compilation flags:
`static`

Template:
`precision(Precision,LatitudeError,LongitudeError)`
Mode and number of proofs:
`precision(+positive_integer,-float,-float)` - `zero_or_one`

---

cell_dimensions/3

Returns the latitude and longitude spans in degrees of a geohash cell for the given precision in characters.

Compilation flags:
`static`

Template:
`cell_dimensions(Precision,LatitudeSpan,LongitudeSpan)`
Mode and number of proofs:
`cell_dimensions(+positive_integer,-float,-float)` - `zero_or_one`

---

adjacent/3

Returns the adjacent geohash in the given direction. Supported directions are `north`, `south`, `east`, `west`, `north_east`, `north_west`, `south_east`, and `south_west`. Eastward and westward adjacency wraps across the antimeridian using the same longitude canonicalization as `encode/3`. Northward and southward adjacency fails when there is no neighboring cell beyond the poles.

Compilation flags:
`static`

Template:
`adjacent(Geohash,Direction,AdjacentGeohash)`
Mode and number of proofs:
`adjacent(+atom,+atom,-atom)` - `zero_or_one`

---

neighbors/2

Returns the available neighboring geohashes as `Direction-Geohash` pairs in clockwise order starting at north.

Compilation flags:
`static`

Template:
`neighbors(Geohash,Neighbors)`
Mode and number of proofs:
`neighbors(+atom,-list(compound))` - `zero_or_one`

---

covering/3

Returns the list of geohashes at the given precision whose cells cover a geographic bounding box represented as `bbox(geographic(MinLatitude,MinLongitude),geographic(MaxLatitude,MaxLongitude))`. Degenerate point bounding boxes and exact cell-aligned boundaries are accepted. Antimeridian-crossing bounding boxes are accepted.

Compilation flags:
`static`

Template:
`covering(BoundingBox,Precision,Geohashes)`
Mode and number of proofs:
`covering(+compound,+positive_integer,-list(atom))` - `zero_or_one`

---

covering/4

Returns a geohash cover for a geographic bounding box. `CoverSpec` is either `precision(Precision)` for a fixed-depth cover or `max_precision(MaxPrecision)` for an adaptive mixed-depth cover. `Options` may include `compact(Boolean)` and `min_precision(PositiveInteger)`.

Compilation flags:
`static`

Template:
`covering(BoundingBox,CoverSpec,Geohashes,Options)`
Mode and number of proofs:
`covering(+compound,+compound,-list(atom),++list(compound))` - `zero_or_one_or_error`

Exceptions:
`Options` or an element of `Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`

An element `Option` of `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of `Options` is not a valid bounding-box covering option:
`domain_error(option,Option)`

---

compress/2

Returns sorted geohashes without duplicates or covered descendants, replacing complete sibling sets with their parent prefix while preserving the covered area.

Compilation flags:
`static`

Template:
`compress(Geohashes,CompressedGeohashes)`
Mode and number of proofs:
`compress(+list(atom),-list(atom))` - `zero_or_one`

---

parent/2

Returns the immediate parent prefix of a geohash. Fails for precision-1 geohashes.

Compilation flags:
`static`

Template:
`parent(Geohash,Parent)`
Mode and number of proofs:
`parent(+atom,-atom)` - `zero_or_one`

---

children/2

Returns the 32 child geohashes of a geohash in standard geohash base-32 order.

Compilation flags:
`static`

Template:
`children(Geohash,Children)`
Mode and number of proofs:
`children(+atom,-list(atom))` - `zero_or_one`

---

common_prefix/3

Returns the longest common geohash prefix shared by two geohashes. The prefix may be the empty atom when the geohashes share no leading characters.

Compilation flags:

```
static
```

Template:

```
common_prefix(Geohash1,Geohash2,Prefix)
```

Mode and number of proofs:

```
common_prefix(+atom,+atom,-atom) - zero_or_one
```

---

encode_int/3

Encodes a geographic coordinate into an integer geohash representation at the given precision in characters.

Compilation flags:

```
static
```

Template:

```
encode_int(Coordinate,Precision,HashInteger)
```

Mode and number of proofs:

```
encode_int(+compound,+positive_integer,-integer) - zero_or_one
```

---

decode_int/3

Decodes an integer geohash representation at the given precision into the center geographic coordinate of the represented cell.

Compilation flags:

```
static
```

Template:

```
decode_int(HashInteger,Precision,Coordinate)
```

Mode and number of proofs:

```
decode_int(+integer,+positive_integer,-compound) - zero_or_one
```

---

bounding_box_int/3

Decodes an integer geohash representation at the given precision into its geographic bounding box.

Compilation flags:
`static`

Template:
`bounding_box_int(HashInteger,Precision,BoundingBox)`
Mode and number of proofs:
`bounding_box_int(+integer,+positive_integer,-compound)` - `zero_or_one`

---

adjacent_int/4

Returns the adjacent integer geohash at the same precision in the given direction.

Compilation flags:
`static`

Template:
`adjacent_int(HashInteger,Precision,Direction,AdjacentHashInteger)`
Mode and number of proofs:
`adjacent_int(+integer,+positive_integer,+atom,-integer)` - `zero_or_one`

---

neighbors_int/3

Returns the available neighboring integer geohashes as `Direction-HashInteger` pairs in clockwise order starting at north.

Compilation flags:
`static`

Template:
`neighbors_int(HashInteger,Precision,Neighbors)`
Mode and number of proofs:
`neighbors_int(+integer,+positive_integer,-list(compound))` - `zero_or_one`

---

geohash_to_int/2

Converts a geohash atom to its packed integer representation using five bits per character.

Compilation flags:
`static`

Template:
`geohash_to_int(Geohash,HashInteger)`
Mode and number of proofs:
`geohash_to_int(+atom,-integer)` - `zero_or_one`

---

int_to_geohash/3

Converts a packed integer geohash representation and an explicit precision in characters to a geohash atom.

Compilation flags:
`static`

Template:
`int_to_geohash(HashInteger,Precision,Geohash)`
Mode and number of proofs:
`int_to_geohash(+integer,+positive_integer,-atom)` - `zero_or_one`

---

encode_bits/3

Encodes a geographic coordinate into an integer geohash representation using the given bit precision.

Compilation flags:
`static`

Template:
`encode_bits(Coordinate,Bits,HashInteger)`
Mode and number of proofs:
`encode_bits(+compound,+positive_integer,-integer)` - `zero_or_one`

---

decode_bits/3

Decodes an integer geohash representation at the given bit precision into the center geographic coordinate of the represented cell.

Compilation flags:
`static`

Template:
`decode_bits(HashInteger,Bits,Coordinate)`
Mode and number of proofs:
`decode_bits(+integer,+positive_integer,-compound)` - `zero_or_one`

---

bounding_box_bits/3

Decodes an integer geohash representation at the given bit precision into its geographic bounding box.

Compilation flags:
`static`

Template:
`bounding_box_bits(HashInteger,Bits,BoundingBox)`
Mode and number of proofs:
`bounding_box_bits(+integer,+positive_integer,-compound)` - `zero_or_one`

---

adjacent_bits/4

Returns the adjacent integer geohash at the same bit precision in the given direction.

Compilation flags:
`static`

Template:
`adjacent_bits(HashInteger,Bits,Direction,AdjacentHashInteger)`
Mode and number of proofs:
`adjacent_bits(+integer,+positive_integer,+atom,-integer)` - `zero_or_one`

---

neighbors_bits/3

Returns the available neighboring integer geohashes as `Direction-HashInteger` pairs in clockwise order starting at north for the given bit precision.

Compilation flags:

```
static
```

Template:

```
neighbors_bits(HashInteger,Bits,Neighbors)
```

Mode and number of proofs:

```
neighbors_bits(+integer,+positive_integer,-list(compound)) - zero_or_one
```

---

expand/2

Returns the geohash followed by its available neighbors in clockwise order starting at north.

Compilation flags:

```
static
```

Template:

```
expand(Geohash,ExpandedGeohashes)
```

Mode and number of proofs:

```
expand(+atom,-list(atom)) - zero_or_one
```

---

expand_int/3

Returns the integer geohash followed by its available neighboring integer geohashes in clockwise order starting at north.

Compilation flags:

```
static
```

Template:

```
expand_int(HashInteger,Precision,ExpandedHashes)
```

Mode and number of proofs:

```
expand_int(+integer,+positive_integer,-list(integer)) - zero_or_one
```

---

polygon_covering/4

Returns a geohash cover for a polygon represented as a list of geographic coordinates. `CoverSpec` is either `precision(Precision)` for a fixed-depth cover or `max_precision(MaxPrecision)` for an adaptive mixed-depth cover. `Options` may include `compact(Boolean)` and `min_precision(PositiveInteger)`. Antimeridian-crossing polygons are not supported.

Compilation flags:

`static`

Template:

`polygon_covering(Polygon,CoverSpec,Geohashes,Options)`

Mode and number of proofs:

`polygon_covering(+list(compound),+compound,-list(atom),++list(compound))` - `zero_or_one_or_error`

Exceptions:

`Options` or an element of `Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of `Options` is not a valid polygon covering option:

`domain_error(option,Option)`

---

polyline_covering/4

Returns a geohash cover for a polyline represented as a list of two or more geographic coordinates. `CoverSpec` is either `precision(Precision)` for a fixed-depth cover or `max_precision(MaxPrecision)` for an adaptive mixed-depth cover. `Options` may include `compact(Boolean)`, `min_precision(PositiveInteger)`, and `buffer(Distance)` where the buffer distance is given in kilometers. Antimeridian-crossing polylines are not supported.

Compilation flags:

`static`

Template:

`polyline_covering(Polyline,CoverSpec,Geohashes,Options)`

Mode and number of proofs:

`polyline_covering(+list(compound),+compound,-list(atom),++list(compound))` - `zero_or_one_or_error`

Exceptions:

`Options` or an element of `Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`
An element `Option` of `Options` is not a valid polyline covering option:
`domain_error(option,Option)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

geohash, *geospatial*

1.99 geojson

object

1.99.1 `geojson`

GeoJSON parser, generator, and validator using default JSON term representations.

Availability:
`logtalk_load(geojson(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-08

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *geojson(curly,dash,atom)*

Remarks:
(none)

Inherited public predicates:
generate/2 *geojson_to_json/2* *json_to_geojson/2* *parse/2* *validate/1* *validate/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.99.2 `geojson(ObjectRepresentation,PairRepresentation,StringRepresentation)`

- `ObjectRepresentation` - Object representation to be used when decoding embedded JSON objects. Possible values are `curly` (default) and `list`.
- `PairRepresentation` - Pair representation to be used when decoding embedded JSON objects. Possible values are `dash` (default), `equal`, and `colon`.
- `StringRepresentation` - Text representation to be used when decoding JSON strings. Possible values are `atom` (default), `chars`, and `codes`.

GeoJSON (RFC 7946) parser, generator, and validator built on top of the `json` library.

Availability:

`logtalk_load(geojson(loader))`

Author: Paulo Moura
Version: 1:0:2
Date: 2026-08-02

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *geojson_protocol*

Uses:

json(ObjectRepresentation,PairRepresentation,StringRepresentation)
list

Remarks:
(none)

Inherited public predicates:
generate/2 *geojson_to_json/2* *json_to_geojson/2* *parse/2* *validate/1* *validate/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

json, *geospatial*

protocol

1.99.3 `geojson_protocol`

GeoJSON (RFC 7946) parser, generator, and validator protocol.

Availability:
`logtalk_load(geojson(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-15

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
  - *validate/1*
  - *validate/2*
  - *json_to_geojson/2*
  - *geojson_to_json/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a GeoJSON document from the given source (`file(Path)`, `stream(Stream)`, `codes(List)`, `chars(List)`, or `atom(Atom)`) into a native GeoJSON term.

Compilation flags:
`static`

Template:
`parse(Source,Term)`

Mode and number of proofs:
`parse(++compound,--term)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(geojson_source,Source)`
Parsed JSON does not represent a valid GeoJSON document:
`domain_error(geojson,Document)`

generate/2

Generates GeoJSON content using the representation specified in the first argument (`file(Path)`, `stream(Stream)`, `codes(List)`, `chars(List)`, or `atom(Atom)`) from a native GeoJSON term.

Compilation flags:
`static`

Template:
`generate(Sink,Term)`
Mode and number of proofs:
`generate(+compound,++term)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid sink:
`domain_error(geojson_sink,Sink)`
`Term` is not a valid GeoJSON term:
`domain_error(geojson,Term)`

---

validate/1

True iff the argument is a valid native GeoJSON term according to RFC 7946.

Compilation flags:
`static`

Template:
`validate(Term)`
Mode and number of proofs:
`validate(@term)` - `zero_or_one`

---

validate/2

Validates a native GeoJSON term returning a list of reason terms whose final argument is the failing path.

Compilation flags:
`static`

Template:
`validate(Term,Errors)`
Mode and number of proofs:
`validate(@term,-list(compound))` - `one_or_error`

Exceptions:

`Term` is a variable:

`instantiation_error`

Remarks:

- invalid_geojson_term(Path): The term is not any supported GeoJSON geometry, feature, or feature collection representation.
- invalid_geometry(Path): A geometry member is neither a valid geometry term nor `@null` where `@null` is allowed.
- invalid_properties(Path): A feature properties member is neither `@null` nor a valid JSON object term in the selected representation.
- invalid_options(Path): The options argument is not a list.
- option_not_allowed(Name, Path): An option with the given name is not allowed for the enclosing GeoJSON term.
- duplicate_option(Name, Path): The same option name occurs more than once in the options list.
- unknown_option(Option, Path): An option term is not recognized as one of the supported options.
- invalid_bbox(Path): A bounding box is not a list of four or six numbers.
- bbox_longitude_out_of_range(Path): A bounding box longitude value lies outside the RFC 7946 range `[-180,180]`.
- bbox_latitude_out_of_range(Path): A bounding box latitude value lies outside the RFC 7946 range `[-90,90]`.
- bbox_latitude_order(Path): The south latitude value is greater than the north latitude value in a bounding box.
- bbox_altitude_order(Path): The minimum altitude value is greater than the maximum altitude value in a three-dimensional bounding box.
- bbox_dimension_mismatch(Actual, Expected, Path): A bounding box length does not match the dimensionality required by the represented geometry, feature, or collection.
- invalid_id(Path): A feature identifier is neither a number nor a valid string term in the selected representation.
- invalid_foreign_members(Path): Foreign members are not represented as a valid pair list or embedded JSON object term, or contain an invalid pair term.
- prohibited_member(Key, Path): A prohibited foreign member name such as `crs` was used.
- reserved_foreign_member(Key, Path): A reserved GeoJSON member name was used as a foreign member key.
- invalid_position(Path): A position is not a list of at least two numeric coordinates.
- position_longitude_out_of_range(Path): A position longitude value lies outside the RFC 7946 range `[-180,180]`.
- position_latitude_out_of_range(Path): A position latitude value lies outside the RFC 7946 range `[-90,90]`.
- invalid_json_object_pair(Path): A JSON object member is not represented using a supported key-value pair term.
- invalid_json_key(Path): A JSON object key is not a valid string term in the selected representation.
- duplicate_json_object_key(Key, Path): The same JSON object key occurs more than once in the same object.
- invalid_json_value(Path): A JSON value is not a number, boolean, null, valid string term, array, or object term.
- insufficient_positions(MinimumLength, Path): A coordinate array contains fewer positions than required for the enclosing geometry.
- invalid_position_array(Path): A coordinate array expected to contain positions is not a list of valid positions.
- inconsistent_position_dimension(Path): Positions in the same coordinate array do not all have the same dimensionality.
- invalid_line_string_array(Path): A MultiLineString coordinate array is not a valid list of line string coordinate arrays.
- invalid_polygon(Path): A polygon coordinate structure is invalid, for example because it is not a non-empty list of linear rings.
- ring_not_closed(Path): A linear ring does not repeat its first position as its last position.
- invalid_multi_polygon(Path): A MultiPolygon coordinate array is not a valid list of polygon coordinate arrays.
- invalid_geometry_collection(Path): A GeometryCollection geometries member is not a list.
- invalid_feature_collection(Path): A FeatureCollection features member is not a list.
- invalid_feature(Path): A value inside a FeatureCollection is not a valid feature term.

json_to_geojson/2

Converts a JSON term, as returned by the `json` library, into a native GeoJSON term.

Compilation flags:
`static`

Template:
`json_to_geojson(JSON,GeoJSON)`
Mode and number of proofs:
`json_to_geojson(+term,-term)` - `one_or_error`

Exceptions:
`JSON` is a variable:
`instantiation_error`
`JSON` is not a valid GeoJSON JSON term:
`domain_error(geojson,JSON)`

---

geojson_to_json/2

Converts a native GeoJSON term into a JSON term suitable for the `json` library.

Compilation flags:
`static`

Template:
`geojson_to_json(GeoJSON,JSON)`
Mode and number of proofs:
`geojson_to_json(+term,-term)` - `one_or_error`

Exceptions:
`GeoJSON` is a variable:
`instantiation_error`
`GeoJSON` is not a valid GeoJSON term:
`domain_error(geojson,GeoJSON)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

geojson, *json*, *geospatial*

1.100 geospatial

object

1.100.1 `geospatial`

Geospatial predicates over geographic coordinates represented as `geographic(Latitude,Longitude)`.

Availability:
`logtalk_load(geospatial(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-10-11

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *geospatial_protocol*
Uses:
list

Remarks:

- Distance unit: Kilometers.
- Coordinate ranges: Latitude values must be in the `[-90.0,90.0]` range and longitude values in the `[-180.0,180.0]` range.

Inherited public predicates:
along_track_distance/4 *bbox_contains/2* *bbox_contains_polygon/2* *bbox_expand/3* *bbox_from_coordinates/2* *bbox_intersects/2* *bbox_intersects_polygon/2* *bbox_intersects_polyline/2* *bbox_overlaps/2* *bbox_union/3* *bounding_box/3* *clockwise_polygon/2* *close_polygon/2* *coordinates_bounding_box/2* *counterclockwise_polygon/2* *cross_track_distance/4* *destination_point/4* *distance/4* *distance/5* *equirectangular_inverse/4* *equirectangular_projection/4* *final_bearing/3* *haversine_distance/3* *initial_bearing/3* *interpolate_great_circle/4* *interpolate_rhumb/4* *is_clockwise_polygon/1* *is_valid_polygon/1* *mean_center/2* *midpoint/3* *minimum_enclosing_circle/3* *nearest_coordinate/5* *nearest_point_on_polyline/4* *nearest_point_on_segment/4* *normalize_coordinate/2* *normalize_polygon_orientation/3* *point_in_polygon/2* *point_to_polyline_distance/3* *polygon_area/2* *polygon_bounding_box/2* *polygon_centroid/2* *polygon_orientation/2* *polygon_perimeter/2* *polygon_perimeter/3* *polygons_intersect/2* *polyline_length/2* *polyline_length/3*

polyline_resample/3 *polyline_simplify/3* *polyline_split_at_distance/4* *rhumb_bearing/3* *rhumb_destination_point/4* *rhumb_distance/3* *rhumb_midpoint/3* *route_distance/2* *route_distance/3* *route_distance/4* *valid_coordinate/1* *vincenty_distance/3* *within_distance/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

geospatial_protocol, *numberlist*

protocol

1.100.2 `geospatial_protocol`

Geospatial predicates protocol.

Availability:
`logtalk_load(geospatial(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-05-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *valid_coordinate/1*
  - *normalize_coordinate/2*
  - *equirectangular_projection/4*
  - *equirectangular_inverse/4*
  - *haversine_distance/3*
  - *vincenty_distance/3*
  - *rhumb_distance/3*
  - *rhumb_bearing/3*
  - *rhumb_destination_point/4*
  - *interpolate_rhumb/4*
  - *rhumb_midpoint/3*
  - *distance/4*
  - *distance/5*
  - *initial_bearing/3*
  - *final_bearing/3*
  - *midpoint/3*
  - *destination_point/4*
  - *interpolate_great_circle/4*
  - *cross_track_distance/4*
  - *along_track_distance/4*
  - *within_distance/4*
  - *nearest_coordinate/5*
  - *mean_center/2*
  - *minimum_enclosing_circle/3*
  - *coordinates_bounding_box/2*
  - *point_in_polygon/2*
  - *polygon_area/2*
  - *polygon_centroid/2*
  - *polygon_bounding_box/2*
  - *close_polygon/2*
  - *polygon_orientation/2*
  - *is_clockwise_polygon/1*
  - *normalize_polygon_orientation/3*
  - *clockwise_polygon/2*
  - *counterclockwise_polygon/2*
  - *is_valid_polygon/1*
  - *bbox_contains/2*
  - *bbox_intersects/2*
  - *bbox_overlaps/2*
  - *bbox_union/3*
  - *bbox_expand/3*
  - *bbox_intersects_polygon/2*
  - *bbox_contains_polygon/2*

- bbox_from_coordinates/2
- bbox_intersects_polyline/2
- point_to_polyline_distance/3
- nearest_point_on_segment/4
- nearest_point_on_polyline/4
- polyline_length/2
- polyline_length/3
- polyline_simplify/3
- polyline_split_at_distance/4
- polyline_resample/3
- polygon_perimeter/2
- polygon_perimeter/3
- polygons_intersect/2
- bounding_box/3
- route_distance/2
- route_distance/3
- route_distance/4

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

valid_coordinate/1

True if the argument is a valid geographic coordinate represented as `geographic(Latitude,Longitude)` with latitude in the `[-90.0,90.0]` range and longitude in the `[-180.0,180.0]` range.

Compilation flags:
`static`

Template:
`valid_coordinate(Coordinate)`

Mode and number of proofs:
`valid_coordinate(@compound)` - `zero_or_one`

---

normalize_coordinate/2

Normalizes a coordinate by wrapping longitude to the `[-180.0,180.0]` range and reflecting latitude over the poles when necessary.

Compilation flags:
`static`

Template:
`normalize_coordinate(Coordinate,NormalizedCoordinate)`

Mode and number of proofs:

```
normalize_coordinate(+compound,-compound) - one
```

---

equirectangular_projection/4

Projects a coordinate to local equirectangular planar coordinates in kilometers using a reference latitude in degrees.

Compilation flags:

```
static
```

Template:

```
equirectangular_projection(Coordinate,ReferenceLatitude,X,Y)
```

Mode and number of proofs:

```
equirectangular_projection(+compound,+float,-float,-float) - zero_or_one
```

---

equirectangular_inverse/4

Converts local equirectangular planar coordinates in kilometers to a geographic coordinate using a reference latitude in degrees.

Compilation flags:

```
static
```

Template:

```
equirectangular_inverse(X,Y,ReferenceLatitude,Coordinate)
```

Mode and number of proofs:

```
equirectangular_inverse(+float,+float,+float,-compound) - zero_or_one
```

---

haversine_distance/3

Computes the great-circle distance in kilometers between two coordinates using the Haversine formula.

Compilation flags:

```
static
```

Template:

```
haversine_distance(Coordinate1,Coordinate2,Distance)
```

Mode and number of proofs:

```
haversine_distance(+compound,+compound,-float) - zero_or_one
```

---

vincenty_distance/3

Computes the geodesic distance in kilometers between two coordinates using the Vincenty inverse formula over the WGS84 ellipsoid. Fails if the iterative method does not converge.

Compilation flags:
`static`

Template:
`vincenty_distance(Coordinate1,Coordinate2,Distance)`
Mode and number of proofs:
`vincenty_distance(+compound,+compound,-float)` - `zero_or_one`

---

rhumb_distance/3

Computes the rhumb-line (loxodrome) distance in kilometers between two coordinates on a spherical Earth model.

Compilation flags:
`static`

Template:
`rhumb_distance(Coordinate1,Coordinate2,Distance)`
Mode and number of proofs:
`rhumb_distance(+compound,+compound,-float)` - `zero_or_one`

---

rhumb_bearing/3

Computes the rhumb-line initial bearing in degrees in the `[0.0,360.0[` range from the first coordinate to the second coordinate.

Compilation flags:
`static`

Template:
`rhumb_bearing(Coordinate1,Coordinate2,Bearing)`
Mode and number of proofs:
`rhumb_bearing(+compound,+compound,-float)` - `zero_or_one`

---

rhumb_destination_point/4

Computes the destination coordinate from a start coordinate, a rhumb-line bearing in degrees, and a distance in kilometers on a spherical Earth model.

Compilation flags:
`static`

Template:
`rhumb_destination_point(Start,Bearing,Distance,Destination)`
Mode and number of proofs:
`rhumb_destination_point(+compound,+float,+float,-compound)` - `zero_or_one`

---

interpolate_rhumb/4

Computes an intermediate coordinate along the rhumb-line path between two coordinates using a fraction in the `[0.0,1.0]` range.

Compilation flags:
`static`

Template:
`interpolate_rhumb(Coordinate1,Coordinate2,Fraction,Coordinate)`
Mode and number of proofs:
`interpolate_rhumb(+compound,+compound,+float,-compound)` - `zero_or_one`

---

rhumb_midpoint/3

Computes the midpoint along the rhumb-line path between two coordinates.

Compilation flags:
`static`

Template:
`rhumb_midpoint(Coordinate1,Coordinate2,Midpoint)`
Mode and number of proofs:
`rhumb_midpoint(+compound,+compound,-compound)` - `zero_or_one`

---

distance/4

Computes the distance in kilometers between two coordinates using a selected metric. Supported metrics are `haversine`, `vincenty`, and `rhumb`.

Compilation flags:

`static`

Template:

`distance(Coordinate1,Coordinate2,Metric,Distance)`

Mode and number of proofs:

`distance(+compound,+compound,+atom,-float)` - `zero_or_one`

---

distance/5

Computes the distance between two coordinates using a selected metric and output unit. Supported metrics are `haversine`, `vincenty`, and `rhumb`. Valid `Unit` argument values are `kilometers`, `meters`, `miles`, and `nautical_miles`.

Compilation flags:

`static`

Template:

`distance(Coordinate1,Coordinate2,Metric,Unit,Distance)`

Mode and number of proofs:

`distance(+compound,+compound,+atom,+atom,-float)` - `zero_or_one`

---

initial_bearing/3

Computes the initial bearing in degrees in the `[0.0,360.0[` range from the first coordinate to the second coordinate.

Compilation flags:

`static`

Template:

`initial_bearing(Coordinate1,Coordinate2,Bearing)`

Mode and number of proofs:

`initial_bearing(+compound,+compound,-float)` - `zero_or_one`

---

final_bearing/3

Computes the final bearing in degrees in the `[0.0,360.0[` range when arriving at the second coordinate from the first coordinate.

Compilation flags:
`static`

Template:
`final_bearing(Coordinate1,Coordinate2,Bearing)`
Mode and number of proofs:
`final_bearing(+compound,+compound,-float)` - `zero_or_one`

---

midpoint/3

Computes the geographic midpoint between two coordinates using a spherical Earth model.

Compilation flags:
`static`

Template:
`midpoint(Coordinate1,Coordinate2,Midpoint)`
Mode and number of proofs:
`midpoint(+compound,+compound,-compound)` - `zero_or_one`

---

destination_point/4

Computes the destination coordinate from a start coordinate, an initial bearing in degrees, and a distance in kilometers using a spherical Earth model.

Compilation flags:
`static`

Template:
`destination_point(Start,Bearing,Distance,Destination)`
Mode and number of proofs:
`destination_point(+compound,+float,+float,-compound)` - `zero_or_one`

---

interpolate_great_circle/4

Computes an intermediate coordinate along the great-circle path between two coordinates using a fraction in the `[0.0,1.0]` range.

Compilation flags:
`static`

Template:
`interpolate_great_circle(Coordinate1,Coordinate2,Fraction,Coordinate)`
Mode and number of proofs:
`interpolate_great_circle(+compound,+compound,+float,-compound)` - `zero_or_one`

---

cross_track_distance/4

Computes the signed cross-track distance in kilometers from a coordinate to the great-circle path defined by a start and end coordinate.

Compilation flags:
`static`

Template:
`cross_track_distance(Coordinate,Start,End,Distance)`
Mode and number of proofs:
`cross_track_distance(+compound,+compound,+compound,-float)` - `zero_or_one`

---

along_track_distance/4

Computes the along-track distance in kilometers from the start coordinate to the closest point to a coordinate on the great-circle path defined by a start and end coordinate.

Compilation flags:
`static`

Template:
`along_track_distance(Coordinate,Start,End,Distance)`
Mode and number of proofs:
`along_track_distance(+compound,+compound,+compound,-float)` - `zero_or_one`

---

within_distance/4

True when the distance between two coordinates is less than or equal to the given radius in kilometers using the selected metric.

Compilation flags:
```
static
```

Template:
```
within_distance(Coordinate1,Coordinate2,Radius,Metric)
```
Mode and number of proofs:
```
within_distance(+compound,+compound,+float,+atom) - zero_or_one
```

---

nearest_coordinate/5

Finds the nearest coordinate to the origin coordinate in a list using the selected metric, returning the nearest coordinate and distance in kilometers.

Compilation flags:
```
static
```

Template:
```
nearest_coordinate(Origin,Coordinates,Metric,Nearest,Distance)
```
Mode and number of proofs:
```
nearest_coordinate(+compound,+list(compound),+atom,-compound,-float) - zero_or_one
```

---

mean_center/2

Computes the arithmetic mean center of a list of one or more coordinates.

Compilation flags:
```
static
```

Template:
```
mean_center(Coordinates,Center)
```
Mode and number of proofs:
```
mean_center(+list(compound),-compound) - zero_or_one
```

---

minimum_enclosing_circle/3

Computes an approximate minimum enclosing circle for a list of one or more coordinates, returning the circle center coordinate and radius in kilometers.

Compilation flags:

```
static
```

Template:

```
minimum_enclosing_circle(Coordinates,Center,Radius)
```

Mode and number of proofs:

```
minimum_enclosing_circle(+list(compound),-compound,-float) - zero_or_one
```

---

coordinates_bounding_box/2

Computes the axis-aligned latitude/longitude bounding box for a list of one or more coordinates.

Compilation flags:

```
static
```

Template:

```
coordinates_bounding_box(Coordinates,BoundingBox)
```

Mode and number of proofs:

```
coordinates_bounding_box(+list(compound),-compound) - zero_or_one
```

---

point_in_polygon/2

True when a coordinate is inside (or on the boundary of) a polygon represented as a list of coordinates. Uses a planar ray-casting algorithm over latitude/longitude coordinates.

Compilation flags:

```
static
```

Template:

```
point_in_polygon(Point,Polygon)
```

Mode and number of proofs:

```
point_in_polygon(+compound,+list(compound)) - zero_or_one
```

---

polygon_area/2

Computes an approximate polygon area in square kilometers by projecting coordinates to a local equirectangular plane and applying the shoelace formula.

Compilation flags:
`static`

Template:
`polygon_area(Polygon,Area)`
Mode and number of proofs:
`polygon_area(+list(compound),-float)` - `zero_or_one`

---

polygon_centroid/2

Computes an approximate polygon centroid by using a local equirectangular projection and planar centroid formula.

Compilation flags:
`static`

Template:
`polygon_centroid(Polygon,Centroid)`
Mode and number of proofs:
`polygon_centroid(+list(compound),-compound)` - `zero_or_one`

---

polygon_bounding_box/2

Computes the axis-aligned latitude/longitude bounding box for a polygon represented as a list of coordinates.

Compilation flags:
`static`

Template:
`polygon_bounding_box(Polygon,BoundingBox)`
Mode and number of proofs:
`polygon_bounding_box(+list(compound),-compound)` - `zero_or_one`

---

close_polygon/2

Returns a closed polygon ring by ensuring the first coordinate is repeated at the end of the list.

Compilation flags:
`static`

Template:
`close_polygon(Polygon,ClosedPolygon)`
Mode and number of proofs:
`close_polygon(+list(compound),-list(compound))` - `zero_or_one`

---

polygon_orientation/2

Computes polygon ring orientation as `clockwise` or `counterclockwise` using a local projected signed area approximation.

Compilation flags:
`static`

Template:
`polygon_orientation(Polygon,Orientation)`
Mode and number of proofs:
`polygon_orientation(+list(compound),-atom)` - `zero_or_one`

---

is_clockwise_polygon/1

True when a polygon ring orientation is clockwise.

Compilation flags:
`static`

Template:
`is_clockwise_polygon(Polygon)`
Mode and number of proofs:
`is_clockwise_polygon(+list(compound))` - `zero_or_one`

---

normalize_polygon_orientation/3

Normalizes a polygon ring orientation to `clockwise` or `counterclockwise`.

Compilation flags:
`static`

Template:
`normalize_polygon_orientation(Polygon,Orientation,OrientedPolygon)`
Mode and number of proofs:
`normalize_polygon_orientation(+list(compound),+atom,-list(compound))` - `zero_or_one`

---

clockwise_polygon/2

Returns a polygon ring with clockwise orientation.

Compilation flags:
`static`

Template:
`clockwise_polygon(Polygon,ClockwisePolygon)`
Mode and number of proofs:
`clockwise_polygon(+list(compound),-list(compound))` - `zero_or_one`

---

counterclockwise_polygon/2

Returns a polygon ring with counterclockwise orientation.

Compilation flags:
`static`

Template:
`counterclockwise_polygon(Polygon,CounterclockwisePolygon)`
Mode and number of proofs:
`counterclockwise_polygon(+list(compound),-list(compound))` - `zero_or_one`

---

is_valid_polygon/1

True when a polygon has at least three valid coordinates after normalizing optional closure.

Compilation flags:
`static`

Template:
`is_valid_polygon(Polygon)`
Mode and number of proofs:
`is_valid_polygon(+list(compound))` - `zero_or_one`

---

bbox_contains/2

True when a coordinate or bounding box is inside or on the boundary of a bounding box term `bbox(geographic(MinLatitude,MinLongitude),geographic(MaxLatitude,MaxLongitude))`. Antimeridian-crossing bounding boxes are supported.

Compilation flags:
`static`

Template:
`bbox_contains(BoundingBox,Contained)`
Mode and number of proofs:
`bbox_contains(+compound,+compound)` - `zero_or_one`

---

bbox_intersects/2

True when two bounding boxes intersect or touch. Antimeridian-crossing bounding boxes are supported.

Compilation flags:
`static`

Template:
`bbox_intersects(BoundingBox1,BoundingBox2)`
Mode and number of proofs:
`bbox_intersects(+compound,+compound)` - `zero_or_one`

---

bbox_overlaps/2

True when two bounding boxes have a strict positive-area overlap. Touching only at an edge or corner fails. Antimeridian-crossing bounding boxes are supported.

Compilation flags:
`static`

Template:
`bbox_overlaps(BoundingBox1,BoundingBox2)`
Mode and number of proofs:
`bbox_overlaps(+compound,+compound)` - `zero_or_one`

---

bbox_union/3

Computes the minimal bounding box containing two bounding boxes.

Compilation flags:
`static`

Template:
`bbox_union(BoundingBox1,BoundingBox2,BoundingBox)`
Mode and number of proofs:
`bbox_union(+compound,+compound,-compound)` - `zero_or_one`

---

bbox_expand/3

Expands a bounding box by a distance in kilometers on all sides using a local spherical approximation.

Compilation flags:
`static`

Template:
`bbox_expand(BoundingBox,Distance,ExpandedBoundingBox)`
Mode and number of proofs:
`bbox_expand(+compound,+number,-compound)` - `zero_or_one`

---

bbox_intersects_polygon/2

True when a bounding box intersects a polygon or either contains the other. Antimeridian-crossing bounding boxes are supported.

Compilation flags:
`static`

Template:
`bbox_intersects_polygon(BoundingBox,Polygon)`
Mode and number of proofs:
`bbox_intersects_polygon(+compound,+list(compound))` - `zero_or_one`

---

bbox_contains_polygon/2

True when a bounding box fully contains a polygon, including polygon vertices on the boundary. Antimeridian-crossing bounding boxes are supported.

Compilation flags:
`static`

Template:
`bbox_contains_polygon(BoundingBox,Polygon)`
Mode and number of proofs:
`bbox_contains_polygon(+compound,+list(compound))` - `zero_or_one`

---

bbox_from_coordinates/2

Computes a bounding box from a list of one or more coordinates.

Compilation flags:
`static`

Template:
`bbox_from_coordinates(Coordinates,BoundingBox)`
Mode and number of proofs:
`bbox_from_coordinates(+list(compound),-compound)` - `zero_or_one`

---

bbox_intersects_polyline/2

True when a bounding box intersects or touches a polyline with two or more coordinates. Antimeridian-crossing bounding boxes are supported.

Compilation flags:
`static`

Template:
`bbox_intersects_polyline(BoundingBox,Polyline)`
Mode and number of proofs:
`bbox_intersects_polyline(+compound,+list(compound))` - `zero_or_one`

---

point_to_polyline_distance/3

Computes the minimum distance in kilometers from a coordinate to a polyline with two or more coordinates.

Compilation flags:
`static`

Template:
`point_to_polyline_distance(Point,Polyline,Distance)`
Mode and number of proofs:
`point_to_polyline_distance(+compound,+list(compound),-float)` - `zero_or_one`

---

nearest_point_on_segment/4

Computes the nearest coordinate on a segment to a coordinate using a local equirectangular approximation.

Compilation flags:
`static`

Template:
`nearest_point_on_segment(Point,SegmentStart,SegmentEnd,NearestPoint)`
Mode and number of proofs:
`nearest_point_on_segment(+compound,+compound,+compound,-compound)` - `zero_or_one`

---

nearest_point_on_polyline/4

Computes the nearest coordinate on a polyline to a coordinate and the corresponding distance in kilometers.

Compilation flags:
`static`

Template:
`nearest_point_on_polyline(Point,Polyline,NearestPoint,Distance)`
Mode and number of proofs:
`nearest_point_on_polyline(+compound,+list(compound),-compound,-float)` - `zero_or_one`

---

polyline_length/2

Computes the polyline length in kilometers for a list of two or more coordinates using the default `haversine` metric.

Compilation flags:
`static`

Template:
`polyline_length(Coordinates,Length)`
Mode and number of proofs:
`polyline_length(+list(compound),-float)` - `zero_or_one`

---

polyline_length/3

Computes the polyline length in kilometers for a list of two or more coordinates using the selected metric.

Compilation flags:
`static`

Template:
`polyline_length(Coordinates,Metric,Length)`
Mode and number of proofs:
`polyline_length(+list(compound),+atom,-float)` - `zero_or_one`

---

polyline_simplify/3

Simplifies a polyline using the Douglas-Peucker algorithm with a tolerance in kilometers.

Compilation flags:
`static`

Template:
`polyline_simplify(Coordinates,Tolerance,SimplifiedCoordinates)`
Mode and number of proofs:
`polyline_simplify(+list(compound),+number,-list(compound))` - `zero_or_one`

---

polyline_split_at_distance/4

Splits a polyline at a distance in kilometers from its first coordinate, returning left and right polylines that share the split coordinate.

Compilation flags:
`static`

Template:
`polyline_split_at_distance(Coordinates,Distance,LeftCoordinates,RightCoordinates)`
Mode and number of proofs:
`polyline_split_at_distance(+list(compound),+number,-list(compound),`
`-list(compound))` - `zero_or_one`

---

polyline_resample/3

Resamples a polyline using a fixed step in kilometers, preserving first and last coordinates.

Compilation flags:
`static`

Template:
`polyline_resample(Coordinates,Step,ResampledCoordinates)`
Mode and number of proofs:
`polyline_resample(+list(compound),+number,-list(compound))` - `zero_or_one`

---

polygon_perimeter/2

Computes the polygon perimeter in kilometers using the default `haversine` metric.

Compilation flags:
`static`

Template:
`polygon_perimeter(Polygon,Perimeter)`
Mode and number of proofs:
`polygon_perimeter(+list(compound),-float)` - `zero_or_one`

---

polygon_perimeter/3

Computes the polygon perimeter in kilometers using the selected metric.

Compilation flags:
`static`

Template:
`polygon_perimeter(Polygon,Metric,Perimeter)`
Mode and number of proofs:
`polygon_perimeter(+list(compound),+atom,-float)` - `zero_or_one`

---

polygons_intersect/2

True when two polygons intersect or one polygon is contained in the other.

Compilation flags:
`static`

Template:
`polygons_intersect(Polygon1,Polygon2)`
Mode and number of proofs:
`polygons_intersect(+list(compound),+list(compound))` - `zero_or_one`

---

bounding_box/3

Computes a spherical bounding box around a center coordinate for a given radius in kilometers. The returned bounding box term is `bbox(geographic(MinLatitude,MinLongitude),geographic(MaxLatitude,MaxLongitude))`.

Compilation flags:
`static`

Template:
`bounding_box(Center,Radius,BoundingBox)`
Mode and number of proofs:
`bounding_box(+compound,+positive_number,-compound)` - `zero_or_one`

---

route_distance/2

Computes the route distance in kilometers for a list of two or more coordinates using the default `haversine` metric.

Compilation flags:
`static`

Template:
`route_distance(Coordinates,Distance)`
Mode and number of proofs:
`route_distance(+list(compound),-float)` - `zero_or_one`

---

route_distance/3

Computes the route distance in kilometers for a list of two or more coordinates using the selected metric. Supported metrics are `haversine`, `vincenty`, and `rhumb`.

Compilation flags:
`static`

Template:
`route_distance(Coordinates,Metric,Distance)`
Mode and number of proofs:
`route_distance(+list(compound),+atom,-float)` - `zero_or_one`

---

route_distance/4

Computes the route distance for a list of two or more coordinates using the selected metric and output unit. Supported metrics are haversine, vincenty, and rhumb. Valid Unit argument values are kilometers, meters, miles, and nautical_miles.

Compilation flags:
static

Template:
route_distance(Coordinates,Metric,Unit,Distance)

Mode and number of proofs:
route_distance(+list(compound),+atom,+atom,-float) - zero_or_one

---

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

geospatial, *numberlistp*, *listp*

1.101 git

object

1.101.1 git

Predicates for accessing a git project current branch and latest commit data.

Availability:
logtalk_load(git(loader))

Author: Paulo Moura
Version: 2:3:0
Date: 2026-10-23

Compilation flags:
static, context_switching_calls

Implements:

`public` *git_protocol*

Uses:

os
user

Remarks:

(none)

Inherited public predicates:

branch/2 *commit_author/2* *commit_date/2* *commit_hash/2* *commit_hash_abbreviated/2* *commit_log/3* *commit_message/2* *remote/3* *root/2* *working_tree_clean/1* *working_tree_status/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.101.2 `git_protocol`

Predicates for accessing a git project current branch and latest commit data.

Availability:

`logtalk_load(git(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2026-10-23

Compilation flags:

`static`

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *branch/2*
  - *root/2*
  - *remote/3*
  - *working_tree_clean/1*
  - *working_tree_status/2*
  - *commit_author/2*
  - *commit_date/2*
  - *commit_hash/2*
  - *commit_hash_abbreviated/2*
  - *commit_message/2*
  - *commit_log/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

branch/2

Returns the name of the current git branch. Fails if the directory is not a git repo or a sub-directory of a git repo directory.

Compilation flags:
`static`

Template:
`branch(Directory,Branch)`

Mode and number of proofs:
`branch(+atom,?atom) - zero_or_one`

---

root/2

Returns the absolute file path for the git repo root. Fails if the directory is not a git repo or a sub-directory of a git repo directory.

Compilation flags:

```
static
```

Template:

```
root(Directory,Root)
```

Mode and number of proofs:

```
root(+atom,?atom) - zero_or_one
```

---

remote/3

Returns the git repo remote URI. Fails if the directory is not a git repo or a sub-directory of a git repo directory.

Compilation flags:

```
static
```

Template:

```
remote(Directory,Remote,URI)
```

Mode and number of proofs:

```
remote(+atom,+atom,?atom) - zero_or_one
```

---

working_tree_clean/1

Succeeds if tracked files have no staged or unstaged changes, including submodule changes. Untracked files are ignored. Fails if `git status` fails.

Compilation flags:

```
static
```

Template:

```
working_tree_clean(Directory)
```

Mode and number of proofs:

```
working_tree_clean(+atom) - zero_or_one
```

---

working_tree_status/2

Returns `git status` porcelain version 1 output for tracked changes, including submodule changes. Untracked files are ignored. Fails if `git status` fails.

Compilation flags:

```
static
```

Template:

```
working_tree_status(Directory,Status)
```

Mode and number of proofs:

```
working_tree_status(+atom,-atom) - zero_or_one
```

---

commit_author/2

Returns the latest commit author. Fails if the directory is not a git repo or a sub-directory of a git repo directory.

Compilation flags:

```
static
```

Template:

```
commit_author(Directory,Author)
```

Mode and number of proofs:

```
commit_author(+atom,-atom) - zero_or_one
```

---

commit_date/2

Returns the latest commit date (strict ISO 8601 format). Fails if the directory is not a git repo or a sub-directory of a git repo directory.

Compilation flags:

```
static
```

Template:

```
commit_date(Directory,Date)
```

Mode and number of proofs:

```
commit_date(+atom,-atom) - zero_or_one
```

---

commit_hash/2

Returns the latest commit hash. Fails if the directory is not a git repo or a sub-directory of a git repo directory.

Compilation flags:
`static`

Template:
`commit_hash(Directory,Hash)`
Mode and number of proofs:
`commit_hash(+atom,-atom)` - `zero_or_one`

---

commit_hash_abbreviated/2

Returns the latest commit abbreviated hash. Fails if the directory is not a git repo or a sub-directory of a git repo directory.

Compilation flags:
`static`

Template:
`commit_hash_abbreviated(Directory,Hash)`
Mode and number of proofs:
`commit_hash_abbreviated(+atom,-atom)` - `zero_or_one`

---

commit_message/2

Returns the latest commit message. Fails if the directory is not a git repo or a sub-directory of a git repo directory.

Compilation flags:
`static`

Template:
`commit_message(Directory,Message)`
Mode and number of proofs:
`commit_message(+atom,-atom)` - `zero_or_one`

---

commit_log/3

Returns the git latest commit log output for the given format (see e.g. https://git-scm.com/docs/pretty-formats). Fails if the directory is not a git repo or a sub-directory of a git repo directory.

Compilation flags:
`static`

Template:
`commit_log(Directory,Format,Output)`
Mode and number of proofs:
`commit_log(+atom,+atom,-atom)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.102 glicko2_periodic_ranker

object

1.102.1 glicko2_periodic_ranker

Multi-period Glicko-2 ranker over temporal pairwise game datasets with explicit rating periods and per-game results.

Availability:
`logtalk_load(glicko2_periodic_ranker(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ranking_dataset_common*
`public` *score_ranker_model_common*
`public` *glicko2_common*
Uses:

avltree
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

temporal_pairwise_ranking_dataset_protocol, *ranker_protocol*, *glicko2_ranker*

1.103 glicko2_ranker

object

1.103.1 `glicko2_ranker`

Glicko-2 pairwise preference ranker. Learns one deterministic rating per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol by applying one Glicko-2 rating-period update over the aggregated pairwise outcomes and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:
`logtalk_load(glicko2_ranker(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ranking_dataset_common*
`public` *score_ranker_model_common*

Uses:
avltree
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2*

pairwise_measurement_dataset_declared_items/2 *pairwise_measurement_dataset_items/2*
pairwise_measurement_dataset_measurements/2 *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3*
ranker_options/2 *scores/2* *temporal_pairwise_dataset_connected_components/2*
temporal_pairwise_dataset_declared_items/2 *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3*
temporal_pairwise_dataset_items/2 *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2*
valid_option/1 *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2*
validate_multiplayer_dataset/1 *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2*
validate_pairwise_measurement_dataset/1 *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1*
validate_temporal_pairwise_dataset/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *elo_ranker*, *rank_centrality*

1.104 gpx

object

1.104.1 `gpx`

GPX 1.1 parser, generator, and validator.

Availability:

```
logtalk_load(gpx(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *gpx_protocol*
Uses:
date
list
reader
xml

Remarks:
(none)

Inherited public predicates:
generate/2 *gpx_to_xml/2* *parse/2* *validate/1* *validate/2* *xml_to_gpx/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.104.2 `gpx_protocol`

GPX 1.1 parser, generator, and validator protocol.

Availability:
`logtalk_load(gpx(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
  - *validate/1*
  - *validate/2*
  - *xml_to_gpx/2*
  - *gpx_to_xml/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a GPX 1.1 document from the given source (`file(Path)`, `stream(Stream)`, `codes(List)`, `chars(List)`, or `atom(Atom)`) into a native GPX term.

Compilation flags:
`static`

Template:
`parse(Source,GPX)`

Mode and number of proofs:
`parse(++compound,--compound)` - `one_or_error`

Exceptions:
`Source` is not a ground term:
`instantiation_error`
`Source` is a ground term but not a valid source:
`domain_error(gpx_source,Source)`
The source does not contain a valid GPX 1.1 document:
`domain_error(gpx,Source)`

generate/2

Generates a GPX 1.1 document to the given sink (`file(Path)`, `stream(Stream)`, `codes(List)`, `chars(List)`, or `atom(Atom)`) from a native GPX term.

Compilation flags:

`static`

Template:

`generate(Sink,GPX)`

Mode and number of proofs:

`generate(+compound,++compound)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid sink:

`domain_error(gpx_sink,Sink)`

`GPX` is not a ground term:

`instantiation_error`

`GPX` is ground but not a valid native GPX term:

`domain_error(gpx,GPX)`

---

validate/1

True iff the argument is a valid native GPX 1.1 term.

Compilation flags:

`static`

Template:

`validate(GPX)`

Mode and number of proofs:

`validate(@term)` - `zero_or_one_or_error`

Exceptions:

`GPX` is a variable:

`instantiation_error`

---

validate/2

Validates a native GPX 1.1 term, returning all detected errors as reason terms whose final argument is the failing path. List indexes in paths are zero-based.

Compilation flags:

`static`

Template:

`validate(GPX,Errors)`

Mode and number of proofs:

`validate(@term,-list(compound))` - `one_or_error`

Exceptions:

`GPX` is a variable:

`instantiation_error`

Remarks:

- invalid_gpx_term(Path): The term is not a supported native GPX term.
- invalid_creator(Path): The required creator is not an atom.
- invalid_properties(Path): A properties argument is not a list.
- invalid_list(Path): A repeated GPX value is not represented by a list.
- invalid_value(Value, Path): A value does not satisfy the required GPX type or range.
- invalid_date_time(Value, Path): A value is not an atom in the XML Schema 1.0 `dateTime` lexical space.
- invalid_year(Value, Path): A value is not an atom in the XML Schema 1.0 `gYear` lexical space.
- invalid_bounds(Path): The minimum latitude is greater than the maximum latitude.
- invalid_extensions(Path): The extension nodes are not valid non-GPX namespaced XML elements.
- duplicate_property(Name, Path): A property occurs more than once in the enclosing GPX term.
- unknown_property(Property, Path): The property is not supported by the enclosing GPX term.

---

xml_to_gpx/2

Converts an XML parser document term into a native GPX 1.1 term.

Compilation flags:

`static`

Template:

`xml_to_gpx(XML,GPX)`

Mode and number of proofs:

`xml_to_gpx(+compound,-compound)` - `one_or_error`

Exceptions:

`XML` is not a ground term:

`instantiation_error`

`XML` is a ground term but not a valid GPX 1.1 XML document term:

`domain_error(gpx,XML)`

---

gpx_to_xml/2

Converts a native GPX 1.1 term into an XML parser document term.

Compilation flags:
`static`

Template:
`gpx_to_xml(GPX,XML)`
Mode and number of proofs:
`gpx_to_xml(+compound,-compound)` - `one_or_error`

Exceptions:
`GPX` is not a ground term:
`instantiation_error`
`GPX` is a ground term but not a valid native GPX term:
`domain_error(gpx,GPX)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.105 gradient_boosting_classifier

object

1.105.1 gradient_boosting_classifier

Gradient boosting classifier using multinomial additive models fitted by regression trees to softmax residuals.

Availability:
`logtalk_load(gradient_boosting_classifier(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:

`public` *probabilistic_classifier_common*

Uses:

format
list
numberlist
regression_tree
type

Remarks:

(none)

Inherited public predicates:

check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *regression_tree*, *gradient_boosting_regression*, *adaptive_boosting_classifier*, *random_forest_classifier*

1.106 gradient_boosting_regression

object

1.106.1 `gradient_boosting_regression`

Gradient boosting regression using regression trees as additive base learners fitted to successive residuals.

Availability:
`logtalk_load(gradient_boosting_regression(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *regressor_common*
Uses:
format
list
population
regression_tree
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

linear_regression, *knn_regression*, *regression_tree*, *random_forest_regression*

1.107 grammars

object

1.107.1 `blank_grammars(Format)`

Blank grammars.

Availability:
`logtalk_load(grammars(loader))`

Author: Paulo Moura
Version: 0:4:0
Date: 2025-10-06

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

      - *white_space//0*
      - *white_spaces//0*
      - *space//0*
      - *spaces//0*
      - *tab//0*
      - *tabs//0*
      - *new_line//0*
      - *new_lines//0*
      - *blank//0*
      - *blanks//0*
      - *non_blank//1*
      - *non_blanks//1*
      - *control//0*
      - *controls//0*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

white_space//0

Consumes a single space or tab.

Compilation flags:
`static`

Mode and number of proofs:
`white_space` - `zero_or_one`

---

white_spaces//0

Consumes zero or more spaces and tabs.

Compilation flags:
`static`

Mode and number of proofs:
`white_spaces` - `one`

---

space//0

Consumes a single space.

Compilation flags:
`static`

Mode and number of proofs:
`space` - `zero_or_one`

---

spaces//0

Consumes zero or more spaces.

Compilation flags:
`static`

Mode and number of proofs:
`spaces` - `one`

---

tab//0

Consumes a single tab.

Compilation flags:
`static`

Mode and number of proofs:
`tab` - `zero_or_one`

---

tabs//0

Consumes zero or more tabs.

Compilation flags:
`static`

Mode and number of proofs:
`tabs` - `one`

---

new_line//0

Consumes a single new line.

Compilation flags:
`static`

Mode and number of proofs:
`new_line` - `zero_or_one`

---

new_lines//0

Consumes zero or more new lines.

Compilation flags:
`static`

Mode and number of proofs:
`new_lines` - `one`

---

blank//0

Consumes a single space, tab, vertical tab, line feed, or new line.

Compilation flags:
`static`

Mode and number of proofs:
`blank` - `zero_or_one`

---

blanks//0

Consumes zero or more spaces, tabs, vertical tabs, line feeds, or new lines.

Compilation flags:
`static`

Mode and number of proofs:
`blanks` - `one`

---

non_blank//1

Returns a single non-blank character or character code.

Compilation flags:
`static`

Template:
`non_blank(NonBlank)`
Mode and number of proofs:
`non_blank(-atomic)` - `zero_or_one`

---

non_blanks//1

Returns a (possibly empty) list of non-blank characters or character codes.

Compilation flags:
`static`

Template:
`non_blanks(NonBlanks)`
Mode and number of proofs:
`non_blanks(-list(atomic))` - `one`

---

control//0

Consumes a single control character or character code. Support for the null control character depends on the Prolog backend.

Compilation flags:
`static`

Mode and number of proofs:
`control` - `zero_or_one`

---

controls//0

Consumes zero or more control characters or character codes. Support for the null control character depends on the Prolog backend.

Compilation flags:
`static`

Mode and number of proofs:
`controls` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.107.2 `ip_grammars(Format)`

IP address grammars.

Availability:
`logtalk_load(grammars(loader))`

Author: Paulo Moura
Version: 0:2:0
Date: 2025-10-06

Compilation flags:
`static, context_switching_calls`

Uses:
number_grammars(Format)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *ipv4//1*
  - *ipv6//1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

ipv4//1

Parses an IPv4 network address in the format XXX.XXX.XXX.XXX where each XXX is an octet (i.e., an integer between 0 and 255).

Compilation flags:
`static`

Template:
`ipv4(Octets)`
Mode and number of proofs:
`ipv4(?list(integer))` - `zero_or_one`

---

ipv6//1

Parses an IPv6 network address in the format XXXX.XXXX.XXXX.XXXX.XXXX.XXXX.XXXX.XXXX where each X is a hexadecimal digit.

Compilation flags:
`static`

Template:
`ipv6(HexDigits)`
Mode and number of proofs:
`ipv6(?list(integer))` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.107.3 number_grammars(Format)

Number grammars.

Availability:
`logtalk_load(grammars(loader))`

Author: Paulo Moura
Version: 0:3:0
Date: 2025-10-06

Compilation flags:
`static, context_switching_calls`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *bit//1*
  - *bits//1*
  - *digit//1*
  - *digits//1*
  - *hex_digit//1*
  - *hex_digits//1*
  - *natural//1*
  - *integer//1*
  - *float//1*
  - *number//1*
  - *sign//1*
  - *dot//1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

bit//1

Parses a single bit.

Compilation flags:
`static`

Template:
`bit(Bit)`
Mode and number of proofs:
`bit(?integer)` - `zero_or_one`

---

bits//1

Parses a sequence of one or more bits.

Compilation flags:
`static`

Template:
`bits(Bits)`
Mode and number of proofs:
`bits(?list(integer))` - `zero_or_one`

---

digit//1

Parses a single decimal digit.

Compilation flags:
`static`

Template:
`digit(Digit)`
Mode and number of proofs:
`digit(?atomic)` - `zero_or_one`

---

digits//1

Parses a sequence of zero of more digits.

Compilation flags:
`static`

Template:
`digits(Digits)`
Mode and number of proofs:
`digits(?list(atomic))` - `one`

---

hex_digit//1

Parses a single hexa-decimal digit.

Compilation flags:
`static`

Template:
`hex_digit(HexDigit)`
Mode and number of proofs:
`hex_digit(?atomic)` - `zero_or_one`

---

hex_digits//1

Parses a sequence of zero or more hexa-decimal digits.

Compilation flags:
`static`

Template:
`hex_digits(HexDigits)`
Mode and number of proofs:
`hex_digits(?list(atomic))` - `one`

---

natural//1

Parses a natural number (a non signed integer).

Compilation flags:
`static`

Template:
`natural(Natural)`
Mode and number of proofs:
`natural(?non_negative_integer)` - `zero_or_one`

---

integer//1

Parses an integer.

Compilation flags:
`static`

Template:
`integer(Integer)`
Mode and number of proofs:
`integer(?integer)` - `zero_or_one`

---

float//1

Parses a float.

Compilation flags:
`static`

Template:
`float(Float)`
Mode and number of proofs:
`float(?float)` - `zero_or_one`

---

number//1

Parses a number (an integer or a float).

Compilation flags:
`static`

Template:
`number(Number)`
Mode and number of proofs:
`number(?number)` - `zero_or_one`

---

sign//1

Parses a number sign (plus or minus).

Compilation flags:
`static`

Template:
`sign(Sign)`
Mode and number of proofs:
`sign(?atomic)` - `zero_or_one`

---

dot//1

Parses a decimal dot.

Compilation flags:
`static`

Template:
`dot(Dot)`
Mode and number of proofs:
`dot(?atomic)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.107.4 `sequence_grammars`

Sequence grammars.

Availability:
`logtalk_load(grammars(loader))`

Author: Paulo Moura
Version: 0:4:0
Date: 2025-10-06

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *zero_or_more//2*
  - *one_or_more//2*
  - *zero_or_more//1*
  - *one_or_more//1*
  - *zero_or_more//0*
  - *one_or_more//0*
  - *without//2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

zero_or_more//2

Eagerly collect zero or more terminals that satisfy the given closure.

Compilation flags:
```
static
```

Template:
```
zero_or_more(Closure,Terminals)
```
Meta-predicate template:
```
zero_or_more(1,*)
```
Mode and number of proofs:
```
zero_or_more(+callable,-list(atomic)) - one
```

---

one_or_more//2

Eagerly collect one or more terminals that satisfy the given closure.

Compilation flags:
```
static
```

Template:
```
one_or_more(Closure,Terminals)
```
Meta-predicate template:
```
one_or_more(1,*)
```
Mode and number of proofs:
```
one_or_more(+callable,-list(atomic)) - zero_or_one
```

---

zero_or_more//1

Eagerly collect zero or more terminals.

Compilation flags:
```
static
```

Template:
```
zero_or_more(Terminals)
```
Mode and number of proofs:
```
zero_or_more(-list(atomic)) - one
```

---

one_or_more//1

Eagerly collect one or more terminals.

Compilation flags:
`static`

Template:
`one_or_more(Terminals)`
Mode and number of proofs:
`one_or_more(-list(atomic))` - `zero_or_one`

---

zero_or_more//0

Eagerly parse zero or more terminals.

Compilation flags:
`static`

Mode and number of proofs:
`zero_or_more` - `one`

---

one_or_more//0

Eagerly parse one or more terminals.

Compilation flags:
`static`

Mode and number of proofs:
`one_or_more` - `zero_or_one`

---

without//2

Collects input terminals until one of the stop terminals is found. The stop terminals are excluded from the collected terminals.

Compilation flags:
`static`

Template:
`without(StopTerminals,Terminals)`
Mode and number of proofs:
`without(+list(atomic),-list(atomic))` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.108 graphs

category

1.108.1 directed_graph_common

Common directed graph predicates shared by directed graph objects. Uses self-dispatch to call object-specific predicates and avltree for internal bookkeeping.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-25

Compilation flags:
`static`

Implements:
`public` *directed_graph_protocol*
Extends:
`public` *graph_common*
Uses:
avltree
list

Remarks:
(none)

Inherited public predicates:
add_edges/3 *add_vertex/3* *add_vertices/3* *all_pairs_min_paths/2* *all_pairs_min_predecessors/2* *breadth_first_order/3* *cycle/2* *delete_edges/3* *delete_vertex/3* *delete_vertices/3* *depth_first_order/3* *edges/2* *empty/1* *has_cycle/1* *has_path/3* *in_degree/3* *is_acyclic/1* *is_bipartite/1* *is_complete/1* *is_sparse/1* *max_path/5* *min_distances/3* *min_path/5* *min_predecessors/3* *neighbors/3* *new/1* *new/2* *new/3* *number_of_edges/2* *number_of_vertices/2* *out_degree/3* *path/3* *reachable/3* *strongly_connected_components/2* *symmetric_closure/2* *topological_sort/2* *transitive_closure/2* *transpose/2* *vertices/2* *weakly_connected_components/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.108.2 `directed_graph_protocol`

Protocol for directed graph predicates such as transpose, closures, topological sorting, degree queries, and strongly connected components.

Availability:

```
logtalk_load(graphs(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-25

Compilation flags:

```
static
```

Extends:

```
public graph_protocol
```

Remarks:

(none)

Inherited public predicates:

add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 breadth_first_order/3 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edges/2 empty/1 has_path/3 is_bipartite/1 is_complete/1 is_sparse/1 max_path/5 min_distances/3 min_path/5 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 path/3 reachable/3 vertices/2

- *Public predicates*
  - *transpose/2*
  - *transitive_closure/2*
  - *symmetric_closure/2*
  - *topological_sort/2*
  - *in_degree/3*
  - *out_degree/3*
  - *is_acyclic/1*
  - *has_cycle/1*
  - *cycle/2*
  - *strongly_connected_components/2*
  - *weakly_connected_components/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

transpose/2

Unifies `NewGraph` with a graph obtained by reversing all edges.

Compilation flags:
`static`

Template:
`transpose(Graph,NewGraph)`
Mode and number of proofs:
`transpose(+graph,-graph)` - `one`

---

transitive_closure/2

Generates the transitive closure of the graph.

Compilation flags:
`static`

Template:
`transitive_closure(Graph,Closure)`
Mode and number of proofs:
`transitive_closure(+graph,-graph)` - `one`

---

symmetric_closure/2

Generates the symmetric closure of the graph.

Compilation flags:
`static`

Template:
`symmetric_closure(Graph,Closure)`
Mode and number of proofs:
`symmetric_closure(+graph,-graph)` - `one`

---

topological_sort/2

Unifies `Sorted` with a topological sort of the vertices in the graph. Assumes the graph is a DAG.

Compilation flags:
`static`

Template:
`topological_sort(Graph,Sorted)`
Mode and number of proofs:
`topological_sort(+graph,-list(vertex))` - `one`

---

in_degree/3

Unifies `Degree` with the number of incoming edges to `Vertex`. Fails if `Vertex` is not in the graph.

Compilation flags:
`static`

Template:
`in_degree(Vertex,Graph,Degree)`
Mode and number of proofs:
`in_degree(+vertex,+graph,-integer)` - `zero_or_one`

---

out_degree/3

Unifies `Degree` with the number of outgoing edges from `Vertex`. Fails if `Vertex` is not in the graph.

Compilation flags:
`static`

Template:
`out_degree(Vertex,Graph,Degree)`
Mode and number of proofs:
`out_degree(+vertex,+graph,-integer)` - `zero_or_one`

---

is_acyclic/1

True iff the graph is a directed acyclic graph (DAG).

Compilation flags:
`static`

Template:
`is_acyclic(Graph)`
Mode and number of proofs:
`is_acyclic(+graph)` - `zero_or_one`

---

has_cycle/1

True iff the graph contains at least one directed cycle.

Compilation flags:
`static`

Template:
`has_cycle(Graph)`
Mode and number of proofs:
`has_cycle(+graph)` - `zero_or_one`

---

cycle/2

Enumerates directed cycles as lists of vertices where the first and last vertices are the same.

Compilation flags:

```
static
```

Template:

```
cycle(Graph,Cycle)
```

Mode and number of proofs:

```
cycle(+graph,-list(vertex)) - zero_or_more
```

---

strongly_connected_components/2

Computes the strongly connected components of the graph using Tarjan's algorithm. Each component is a sorted list of vertices. Components are returned in topological order.

Compilation flags:

```
static
```

Template:

```
strongly_connected_components(Graph,SCCs)
```

Mode and number of proofs:

```
strongly_connected_components(+graph,-list(list(vertex))) - one
```

---

weakly_connected_components/2

Computes the weakly connected components of the graph by considering edge directions as undirected. Each component is returned as a sorted list of vertices.

Compilation flags:

```
static
```

Template:

```
weakly_connected_components(Graph,Components)
```

Mode and number of proofs:

```
weakly_connected_components(+graph,-list(list(vertex))) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.108.3 `graph_common`

Common graph predicates shared by all graph objects. Uses self-dispatch to call object-specific predicates such as `neighbors/3`, `vertices/2`, and `edges/2`.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-11-16

Compilation flags:
`static`

Implements:
`public` *graph_protocol*
Uses:
list
set

Remarks:
(none)

Inherited public predicates:
add_edges/3 *add_vertex/3* *add_vertices/3* *all_pairs_min_paths/2* *all_pairs_min_predecessors/2* *breadth_first_order/3* *delete_edges/3* *delete_vertex/3* *delete_vertices/3* *depth_first_order/3* *edges/2* *empty/1* *has_path/3* *is_bipartite/1* *is_complete/1* *is_sparse/1* *max_path/5* *min_distances/3* *min_path/5* *min_predecessors/3* *neighbors/3* *new/1* *new/2* *new/3* *number_of_edges/2* *number_of_vertices/2* *path/3* *reachable/3* *vertices/2*

- *Public predicates*
- *Protected predicates*
  - *pairs_to_edges/2*
  - *vertex_neighbors_to_edges/4*
  - *wpairs_to_edges/2*
  - *wvertex_neighbors_to_edges/4*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

pairs_to_edges/2

Converts a list of `Vertex-Neighbors` pairs from a dictionary into a flat list of `Vertex1-Vertex2` edges.

Compilation flags:
`static`

Template:
`pairs_to_edges(Pairs,Edges)`
Mode and number of proofs:
`pairs_to_edges(+list(pair),-list)` - `one`

---

vertex_neighbors_to_edges/4

Converts a neighbor list for a vertex into edges using a difference list.

Compilation flags:
`static`

Template:
`vertex_neighbors_to_edges(Neighbors,Vertex,Edges,RestEdges)`
Mode and number of proofs:
`vertex_neighbors_to_edges(+list,+vertex,-list,-list)` - `one`

---

wpairs_to_edges/2

Converts a list of `Vertex-WNeighbors` pairs from a dictionary into a flat list of `(Vertex1-Vertex2)-Weight` weighted edges.

Compilation flags:
`static`

Template:
`wpairs_to_edges(Pairs,Edges)`
Mode and number of proofs:

```
wpairs_to_edges(+list(pair),-list) - one
```

---

wvertex_neighbors_to_edges/4

Converts a weighted neighbor list for a vertex into weighted edges using a difference list.

Compilation flags:

```
static
```

Template:

```
wvertex_neighbors_to_edges(WNeighbors,Vertex,Edges,RestEdges)
```

Mode and number of proofs:

```
wvertex_neighbors_to_edges(+list,+vertex,-list,-list) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.108.4 graph_protocol

Common protocol for all types of graphs. Graphs are represented using a dictionary where keys are vertices and values are sorted lists of neighbors (implicitly defining edges).

Availability:

```
logtalk_load(graphs(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-25

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *new/1*
  - *new/2*
  - *new/3*
  - *empty/1*
  - *vertices/2*
  - *edges/2*
  - *add_vertex/3*
  - *add_vertices/3*
  - *delete_vertex/3*
  - *delete_vertices/3*
  - *add_edges/3*
  - *delete_edges/3*
  - *neighbors/3*
  - *reachable/3*
  - *breadth_first_order/3*
  - *depth_first_order/3*
  - *number_of_vertices/2*
  - *number_of_edges/2*
  - *path/3*
  - *has_path/3*
  - *min_path/5*
  - *max_path/5*
  - *min_distances/3*
  - *min_predecessors/3*
  - *all_pairs_min_paths/2*
  - *all_pairs_min_predecessors/2*
  - *is_complete/1*
  - *is_bipartite/1*
  - *is_sparse/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new/1

Creates a new empty graph.

Compilation flags:
`static`

Template:
`new(Graph)`

Mode and number of proofs:
`new(-graph) - one`

---

new/2

Creates a new graph from a list of edges. Vertices are defined implicitly by edges.

Compilation flags:
`static`

Template:
`new(Edges,Graph)`
Mode and number of proofs:
`new(+list(edge),-graph)` - `one`

---

new/3

Creates a new graph from a list of vertices and a list of edges. Vertices may also be defined implicitly by edges.

Compilation flags:
`static`

Template:
`new(Vertices,Edges,Graph)`
Mode and number of proofs:
`new(+list(vertex),+list(edge),-graph)` - `one`

---

empty/1

True iff the given graph is empty.

Compilation flags:
`static`

Template:
`empty(Graph)`
Mode and number of proofs:
`empty(@graph)` - `zero_or_one`

---

vertices/2

Unifies `Vertices` with a sorted list of all vertices in the graph.

Compilation flags:
`static`

Template:
`vertices(Graph,Vertices)`
Mode and number of proofs:
`vertices(+graph,-list(vertex))` - `one`

---

edges/2

Unifies `Edges` with a list of all edges in the graph.

Compilation flags:
`static`

Template:
`edges(Graph,Edges)`
Mode and number of proofs:
`edges(+graph,-list(edge))` - `one`

---

add_vertex/3

Adds a vertex to the graph. If the vertex already exists, the graph is unchanged.

Compilation flags:
`static`

Template:
`add_vertex(Graph,Vertex,NewGraph)`
Mode and number of proofs:
`add_vertex(+graph,+vertex,-graph)` - `one`

---

add_vertices/3

Adds a list of vertices to the graph.

Compilation flags:
```
static
```

Template:
```
add_vertices(Graph,Vertices,NewGraph)
```
Mode and number of proofs:
```
add_vertices(+graph,+list(vertex),-graph) - one
```

---

delete_vertex/3

Deletes a vertex and all edges incident to it from the graph. If the vertex does not exist, the graph is unchanged.

Compilation flags:
```
static
```

Template:
```
delete_vertex(Graph,Vertex,NewGraph)
```
Mode and number of proofs:
```
delete_vertex(+graph,+vertex,-graph) - one
```

---

delete_vertices/3

Deletes a list of vertices and all edges incident to them from the graph.

Compilation flags:
```
static
```

Template:
```
delete_vertices(Graph,Vertices,NewGraph)
```
Mode and number of proofs:
```
delete_vertices(+graph,+list(vertex),-graph) - one
```

---

add_edges/3

Adds a list of edges to the graph. Vertices referenced by edges are added implicitly.

Compilation flags:
`static`

Template:
`add_edges(Graph,Edges,NewGraph)`
Mode and number of proofs:
`add_edges(+graph,+list(edge),-graph)` - `one`

---

delete_edges/3

Deletes a list of edges from the graph. Vertices are not deleted. Non-existing edges are silently ignored.

Compilation flags:
`static`

Template:
`delete_edges(Graph,Edges,NewGraph)`
Mode and number of proofs:
`delete_edges(+graph,+list(edge),-graph)` - `one`

---

neighbors/3

Unifies `Neighbors` with a sorted list of the neighbors of `Vertex` in the graph. Fails if `Vertex` is not in the graph.

Compilation flags:
`static`

Template:
`neighbors(Vertex,Graph,Neighbors)`
Mode and number of proofs:
`neighbors(+vertex,+graph,-list(vertex))` - `zero_or_one`

---

reachable/3

Unifies Vertices with a sorted list of vertices reachable from Vertex (including Vertex itself). Fails if Vertex is not in the graph.

Compilation flags:
static

Template:
reachable(Vertex,Graph,Vertices)
Mode and number of proofs:
reachable(+vertex,+graph,-list(vertex)) - zero_or_one

---

breadth_first_order/3

Unifies Vertices with the breadth-first traversal order rooted at Vertex. Fails if Vertex is not in the graph.

Compilation flags:
static

Template:
breadth_first_order(Vertex,Graph,Vertices)
Mode and number of proofs:
breadth_first_order(+vertex,+graph,-list(vertex)) - zero_or_one

---

depth_first_order/3

Unifies Vertices with the depth-first traversal order rooted at Vertex. Fails if Vertex is not in the graph.

Compilation flags:
static

Template:
depth_first_order(Vertex,Graph,Vertices)
Mode and number of proofs:
depth_first_order(+vertex,+graph,-list(vertex)) - zero_or_one

---

number_of_vertices/2

Unifies N with the number of vertices in the graph.

Compilation flags:
```
static
```

Template:
```
number_of_vertices(Graph,N)
```
Mode and number of proofs:
```
number_of_vertices(+graph,-integer) - one
```

---

number_of_edges/2

Unifies N with the number of edges in the graph.

Compilation flags:
```
static
```

Template:
```
number_of_edges(Graph,N)
```
Mode and number of proofs:
```
number_of_edges(+graph,-integer) - one
```

---

path/3

Returns a maximal path (list of vertices) rooted at Vertex, enumerating different paths on backtracking. Fails if Vertex is not in the graph.

Compilation flags:
```
static
```

Template:
```
path(Vertex,Graph,Path)
```
Mode and number of proofs:
```
path(+vertex,+graph,-list(vertex)) - zero_or_more
```

---

has_path/3

True iff there is a path from `Vertex1` to `Vertex2` in the graph.

Compilation flags:
`static`

Template:
`has_path(Vertex1,Vertex2,Graph)`
Mode and number of proofs:
`has_path(+vertex,+vertex,+graph)` - `zero_or_one`

---

min_path/5

Finds the minimum cost path from `Vertex1` to `Vertex2`. For unweighted graphs, cost is the number of edges. Fails if no path exists.

Compilation flags:
`static`

Template:
`min_path(Vertex1,Vertex2,Graph,Path,Cost)`
Mode and number of proofs:
`min_path(+vertex,+vertex,+graph,-list(vertex),-number)` - `zero_or_one`

---

max_path/5

Finds the maximum cost acyclic path from `Vertex1` to `Vertex2`. For unweighted graphs, cost is the number of edges. Fails if no path exists.

Compilation flags:
`static`

Template:
`max_path(Vertex1,Vertex2,Graph,Path,Cost)`
Mode and number of proofs:
`max_path(+vertex,+vertex,+graph,-list(vertex),-number)` - `zero_or_one`

---

min_distances/3

Computes minimum path costs from `Vertex` to all reachable vertices. Returns a list of `Target-Cost` pairs including `Vertex-0`.

Compilation flags:

`static`

Template:

`min_distances(Vertex,Graph,Distances)`

Mode and number of proofs:

`min_distances(+vertex,+graph,-list(pair))` - `zero_or_one`

---

min_predecessors/3

Computes predecessor links for minimum paths rooted at `Vertex`. Returns a list of `Target-Predecessor` pairs; the source predecessor is `none`.

Compilation flags:

`static`

Template:

`min_predecessors(Vertex,Graph,Predecessors)`

Mode and number of proofs:

`min_predecessors(+vertex,+graph,-list(pair))` - `zero_or_one`

---

all_pairs_min_paths/2

Computes minimum path costs for all ordered pairs of vertices with a path between them. Returns a list of `(Vertex1-Vertex2)-Cost` terms.

Compilation flags:

`static`

Template:

`all_pairs_min_paths(Graph,Pairs)`

Mode and number of proofs:

`all_pairs_min_paths(+graph,-list(pair))` - `one`

---

all_pairs_min_predecessors/2

Computes predecessor links for minimum paths for all ordered source-target pairs with a path. Returns a list of `(Vertex1-Vertex2)-Predecessor` terms.

Compilation flags:

`static`

Template:

`all_pairs_min_predecessors(Graph,Pairs)`

Mode and number of proofs:

`all_pairs_min_predecessors(+graph,-list(pair))` - `one`

---

is_complete/1

True iff every pair of distinct vertices in the graph is connected by an edge.

Compilation flags:

`static`

Template:

`is_complete(Graph)`

Mode and number of proofs:

`is_complete(+graph)` - `zero_or_one`

---

is_bipartite/1

True iff the graph is bipartite, i.e. its vertices can be partitioned into two sets such that every edge connects a vertex in one set to a vertex in the other.

Compilation flags:

`static`

Template:

`is_bipartite(Graph)`

Mode and number of proofs:

`is_bipartite(+graph)` - `zero_or_one`

`is_sparse/1`

True iff the graph is sparse, i.e. the number of edges is at most `|V| * log2(|V|)`. The cutoff `|E| = |V| * log2(|V|)` separates sparse graphs (where adjacency lists are efficient) from dense graphs (where adjacency matrix representations may be preferable).

Compilation flags:
`static`

Template:
`is_sparse(Graph)`
Mode and number of proofs:
`is_sparse(+graph)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.108.5 `graph_types`

A set of graph related types and generators.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-02-19

Compilation flags:
`static`

Provides:
type::type/1
type::check/2
arbitrary::arbitrary/1
arbitrary::arbitrary/2
Uses:
integer
type

Remarks:

- Provided types: This category adds `vertex`, `edge`, and `weighted_edge` types for type-checking when using the `type` library object.
- Type `vertex`: Any non-variable term.
- Type `edge`: An unweighted edge represented as a `V1-V2` pair of vertices.
- Type `weighted_edge`: A weighted edge represented as `(V1-V2)-Weight` where `V1` and `V2` are vertices and `Weight` is a number.
- Generating edges: Use the `edges(N,V0,V)` generator for unweighted edges or `weighted_edges(N,V0,V,W)` for weighted edges. `N` is the upper limit to the number of edges. `V0` and `V` must be positive integers and will be used for the range of vertices. `W` is the upper limit for edge weights (positive integers).

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

category

1.108.6 `undirected_graph_common`

Common predicates shared by undirected graph objects. Uses self-dispatch to call object-specific predicates such as `is_connected/1`, `vertices/2`, `edges/2`, and `neighbors/3`.

Availability:

```
logtalk_load(graphs(loader))
```

Author: Paulo Moura
Version: 1:1:1
Date: 2026-11-16

Compilation flags:

`static`

Extends:

`public` *graph_common*

Uses:

avltree
list
set

Remarks:

(none)

Inherited public predicates:

add_edges/3 *add_vertex/3* *add_vertices/3* *all_pairs_min_paths/2* *all_pairs_min_predecessors/2* *breadth_first_order/3* *delete_edges/3* *delete_vertex/3* *delete_vertices/3* *depth_first_order/3* *edges/2* *empty/1* *has_path/3* *is_bipartite/1* *is_complete/1* *is_sparse/1* *max_path/5* *min_distances/3* *min_path/5* *min_predecessors/3* *neighbors/3* *new/1* *new/2* *new/3* *number_of_edges/2* *number_of_vertices/2* *path/3* *reachable/3* *vertices/2*

- *Public predicates*
  - *is_connected/1*
  - *is_tree/1*
  - *has_cycle/1*
  - *cycle/2*
  - *graph_coloring/3*
  - *articulation_points/2*
  - *bridges/2*
  - *maximal_cliques/2*
  - *maximum_cliques/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

is_connected/1

True iff `Graph` is connected (all vertices are reachable from any vertex).

Compilation flags:

`static`

Template:

`is_connected(Graph)`

Mode and number of proofs:

`is_connected(+graph)` - `zero_or_one`

is_tree/1

True iff the graph is a tree, i.e. it is connected and has exactly `|V| - 1` edges.

Compilation flags:
`static`

Template:
`is_tree(Graph)`
Mode and number of proofs:
`is_tree(+graph)` - `zero_or_one`

---

has_cycle/1

True iff the graph contains at least one cycle.

Compilation flags:
`static`

Template:
`has_cycle(Graph)`
Mode and number of proofs:
`has_cycle(+graph)` - `zero_or_one`

---

cycle/2

Enumerates cycles as lists of vertices where the first and last vertices are the same.

Compilation flags:
`static`

Template:
`cycle(Graph,Cycle)`
Mode and number of proofs:
`cycle(+graph,-list(vertex))` - `zero_or_more`

---

graph_coloring/3

Computes a greedy vertex coloring of the graph. `Coloring` is a list of `Vertex-Color` pairs where colors are integers starting from 1. `NumberOfColors` is the total number of colors used.

Compilation flags:
`static`

Template:
`graph_coloring(Graph,Coloring,NumberOfColors)`
Mode and number of proofs:
`graph_coloring(+graph,-list(pair),-integer)` - `one`

---

articulation_points/2

Computes all articulation points (cut vertices) of the graph. The result is a sorted list of vertices.

Compilation flags:
`static`

Template:
`articulation_points(Graph,Points)`
Mode and number of proofs:
`articulation_points(+graph,-list(vertex))` - `one`

---

bridges/2

Computes all bridges (cut edges) of the graph. Each bridge is returned once as `Vertex1-Vertex2` with `Vertex1 @< Vertex2`.

Compilation flags:
`static`

Template:
`bridges(Graph,Bridges)`
Mode and number of proofs:
`bridges(+graph,-list(edge))` - `one`

---

maximal_cliques/2

Computes all maximal cliques of the graph using the Bron-Kerbosch algorithm with pivoting. A maximal clique is a clique that cannot be extended by adding another adjacent vertex. Each clique is a sorted list of vertices. The list of cliques is sorted in standard order.

Compilation flags:

```
static
```

Template:

```
maximal_cliques(Graph,Cliques)
```

Mode and number of proofs:

```
maximal_cliques(+graph,-list(list(vertex)))-one
```

---

maximum_cliques/2

Computes all maximum cliques of the graph, i.e. the largest maximal cliques. Each clique is a sorted list of vertices. The list of cliques is sorted in standard order. For an empty graph, returns an empty list.

Compilation flags:

```
static
```

Template:

```
maximum_cliques(Graph,Cliques)
```

Mode and number of proofs:

```
maximum_cliques(+graph,-list(list(vertex)))-one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.108.7 unweighted_directed_graph

Unweighted directed graph predicates using the AVL tree dictionary representation.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *unweighted_directed_graph(avltree)*

Remarks:
(none)

Inherited public predicates:
add_edge/4 add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 breadth_first_order/3 complement/2 compose/3 cycle/2 delete_edge/4 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edge/3 edges/2 empty/1 has_cycle/1 has_path/3 in_degree/3 is_acyclic/1 is_bipartite/1 is_complete/1 is_sparse/1 leaves/2 max_path/5 min_distances/3 min_path/5 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 out_degree/3 path/3 reachable/3 strongly_connected_components/2 symmetric_closure/2 topological_sort/2 topological_sort/3 transitive_closure/2 transitive_reduction/2 transpose/2 union/3 vertices/2 weakly_connected_components/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.108.8 `unweighted_directed_graph(Dictionary)`

Unweighted directed graph predicates using a dictionary representation. The parametric object parameter is the dictionary to use for the graph representation.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *unweighted_graph_common(Dictionary)*
`public` *directed_graph_common*
Uses:
pairs
set

Remarks:
(none)

Inherited public predicates:
add_edge/4 add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 breadth_first_order/3 complement/2 cycle/2 delete_edge/4 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edge/3 edges/2 empty/1 has_cycle/1 has_path/3 in_degree/3 is_acyclic/1 is_bipartite/1 is_complete/1 is_sparse/1 max_path/5 min_distances/3 min_path/5 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 out_degree/3 path/3 reachable/3 strongly_connected_components/2 symmetric_closure/2 topological_sort/2 transitive_closure/2 transpose/2 vertices/2 weakly_connected_components/2

- *Public predicates*
  - *compose/3*
  - *union/3*
  - *topological_sort/3*
  - *leaves/2*
  - *transitive_reduction/2*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

compose/3

Composes `NewGraph` by connecting the drains of `LeftGraph` to the sources of `RightGraph`.

Compilation flags:
`static`

Template:
`compose(LeftGraph,RightGraph,NewGraph)`
Mode and number of proofs:
`compose(+graph,+graph,-graph)` - `one`

---

union/3

Unifies `UnionGraph` with the union of `Graph1` and `Graph2`.

Compilation flags:
`static`

Template:
`union(Graph1,Graph2,UnionGraph)`
Mode and number of proofs:
`union(+graph,+graph,-graph)` - `one`

---

topological_sort/3

Difference list version of `topological_sort/2` where `Sorted0` is the tail of `Sorted`.

Compilation flags:
`static`

Template:
`topological_sort(Graph,Sorted0,Sorted)`
Mode and number of proofs:
`topological_sort(+graph,+list(vertex),-list(vertex))` - `one`

---

leaves/2

Unifies `Leaves` with a sorted list of vertices with no outgoing edges.

Compilation flags:
`static`

Template:
`leaves(Graph,Leaves)`
Mode and number of proofs:
`leaves(+graph,-list(vertex))` - `one`

---

transitive_reduction/2

Computes the transitive reduction of the graph. An edge Vertex1-Vertex2 is in the reduction iff it is in the graph and there is no other path of length >= 2 from Vertex1 to Vertex2.

Compilation flags:
`static`

Template:
`transitive_reduction(Graph,Reduction)`
Mode and number of proofs:
`transitive_reduction(+graph,-graph)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.108.9 unweighted_graph_common(Dictionary)

Common unweighted graph predicates shared by both unweighted directed and unweighted undirected graph objects. Uses self-dispatch to call object-specific predicates such as `add_edge/4`, `delete_edge/4`, and `neighbors/3`.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-02-20

Compilation flags:
`static`

Implements:
`public` *unweighted_graph_protocol*
Extends:
`public` *graph_common*
Uses:
avltree
list
set

Remarks:
(none)

Inherited public predicates:
add_edge/4 add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 breadth_first_order/3 complement/2 delete_edge/4 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edge/3 edges/2 empty/1 has_path/3 is_bipartite/1 is_complete/1 is_sparse/1 max_path/5 min_distances/3 min_path/5 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 path/3 reachable/3 vertices/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.108.10 unweighted_graph_protocol

Protocol for unweighted graph predicates, extending the common graph protocol with unweighted edge operations.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-19

Compilation flags:
`static`

Extends:
`public` *graph_protocol*

Remarks:
(none)

Inherited public predicates:
add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 breadth_first_order/3 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edges/2 empty/1 has_path/3 is_bipartite/1 is_complete/1 is_sparse/1 max_path/5 min_distances/3 min_path/5 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 path/3 reachable/3 vertices/2

- *Public predicates*
  - *edge/3*
  - *add_edge/4*
  - *delete_edge/4*
  - *complement/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

edge/3

True iff there is an edge between `Vertex1` and `Vertex2` in `Graph`.

Compilation flags:
`static`

Template:
`edge(Vertex1,Vertex2,Graph)`
Mode and number of proofs:
`edge(+vertex,+vertex,+graph)` - `zero_or_one`

---

add_edge/4

Adds an edge between `Vertex1` and `Vertex2` to the graph.

Compilation flags:
`static`

Template:
`add_edge(Graph,Vertex1,Vertex2,NewGraph)`
Mode and number of proofs:
`add_edge(+graph,+vertex,+vertex,-graph)` - `one`

---

delete_edge/4

Deletes the edge between `Vertex1` and `Vertex2` from the graph. The graph is unchanged if the edge does not exist.

Compilation flags:
`static`

Template:
`delete_edge(Graph,Vertex1,Vertex2,NewGraph)`
Mode and number of proofs:
`delete_edge(+graph,+vertex,+vertex,-graph)` - `one`

---

complement/2

Unifies `NewGraph` with the complement graph where there is an edge between all pairs of vertices not connected in the original graph.

Compilation flags:
`static`

Template:
`complement(Graph,NewGraph)`
Mode and number of proofs:
`complement(+graph,-graph)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.108.11 unweighted_undirected_graph

Unweighted undirected graph predicates using the AVL tree dictionary representation.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *unweighted_undirected_graph(avltree)*

Remarks:
(none)

Inherited public predicates:

add_edge/4 *add_edges/3* *add_vertex/3* *add_vertices/3* *all_pairs_min_paths/2* *all_pairs_min_predecessors/2* *articulation_points/2* *breadth_first_order/3* *bridges/2* *complement/2* *connected_components/2* *cycle/2* *degree/3* *delete_edge/4* *delete_edges/3* *delete_vertex/3* *delete_vertices/3* *depth_first_order/3* *edge/3* *edges/2* *empty/1* *graph_coloring/3* *has_cycle/1* *has_path/3* *is_bipartite/1* *is_complete/1* *is_connected/1* *is_sparse/1* *is_tree/1* *max_path/5* *maximal_cliques/2* *maximum_cliques/2* *min_distances/3* *min_path/5* *min_predecessors/3* *neighbors/3* *new/1* *new/2* *new/3* *number_of_edges/2* *number_of_vertices/2* *path/3* *reachable/3* *vertices/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.108.12 `unweighted_undirected_graph(Dictionary)`

Unweighted undirected graph predicates using a dictionary representation. Undirected edges are stored as two directed edges. The parametric object parameter is the dictionary to use for the graph representation.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *unweighted_graph_common(Dictionary)*
`public` *undirected_graph_common*
Uses:
list
set

Remarks:
(none)

Inherited public predicates:
add_edge/4 add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 articulation_points/2 breadth_first_order/3 bridges/2 complement/2 cycle/2 delete_edge/4 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edge/3 edges/2 empty/1 graph_coloring/3 has_cycle/1 has_path/3 is_bipartite/1 is_complete/1 is_connected/1 is_sparse/1 is_tree/1 max_path/5 maximal_cliques/2 maximum_cliques/2 min_distances/3 min_path/5 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 path/3 reachable/3 vertices/2

- *Public predicates*
  - *degree/3*
  - *connected_components/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

degree/3

Unifies `Degree` with the number of edges incident to `Vertex`. Fails if `Vertex` is not in the graph.

Compilation flags:
`static`

Template:
`degree(Vertex,Graph,Degree)`
Mode and number of proofs:
`degree(+vertex,+graph,-integer)` - `zero_or_one`

---

connected_components/2

Unifies `Components` with a list of connected components. Each component is a sorted list of vertices.

Compilation flags:
`static`

Template:
`connected_components(Graph,Components)`
Mode and number of proofs:
`connected_components(+graph,-list(list(vertex)))` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.108.13 `weighted_directed_graph`

Weighted directed graph predicates using the AVL tree dictionary representation. Edge weights use a pair representation (Vertex-Weight in neighbor lists, (Vertex1-Vertex2)-Weight for edge lists).

Availability:

```
logtalk_load(graphs(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-19

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public weighted_directed_graph(avltree)
```

Remarks:

(none)

Inherited public predicates:

add_edge/5 add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 breadth_first_order/3 cycle/2 delete_edge/5 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edge/4 edges/2 empty/1 has_cycle/1 has_negative_cycle/1 has_path/3 in_degree/3 is_acyclic/1 is_bipartite/1 is_complete/1 is_sparse/1 max_path/5 min_distances/3 min_path/5 min_path_bellman_ford/5 min_paths/3 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 out_degree/3 path/3 reachable/3 strongly_connected_components/2 symmetric_closure/2 topological_sort/2 transitive_closure/2 transpose/2 vertices/2 weakly_connected_components/2 wneighbors/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.108.14 `weighted_directed_graph(Dictionary)`

Weighted directed graph predicates using a dictionary representation. Edge weights use a pair representation (Vertex-Weight in neighbor lists, (Vertex1-Vertex2)-Weight for edge lists). The parametric object parameter is the dictionary to use for the graph representation.

Availability:

```
logtalk_load(graphs(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-16

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public weighted_graph_common(Dictionary)
public directed_graph_common
```

Uses:

pairs

Remarks:

(none)

Inherited public predicates:

add_edge/5 add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 breadth_first_order/3 cycle/2 delete_edge/5 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edge/4 edges/2 empty/1 has_cycle/1 has_negative_cycle/1 has_path/3 in_degree/3 is_acyclic/1 is_bipartite/1 is_complete/1 is_sparse/1 max_path/5 min_distances/3 min_path/5 min_path_bellman_ford/5 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 out_degree/3 path/3 reachable/3 strongly_connected_components/2 symmetric_closure/2 topological_sort/2 transitive_closure/2 transpose/2 vertices/2 weakly_connected_components/2 wneighbors/3

- *Public predicates*
  - *min_paths/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

min_paths/3

Computes shortest path tree from `Vertex1` to all reachable vertices.

Compilation flags:
`static`

Template:
`min_paths(Vertex1,Graph,PathTree)`
Mode and number of proofs:
`min_paths(+vertex,+graph,-graph)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.108.15 weighted_graph_common(Dictionary)

Common weighted graph predicates shared by both weighted directed and weighted undirected graph objects. Uses self-dispatch to call object-specific predicates such as `add_edge/5`, `delete_edge/5`, and `edges/2`.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-16

Compilation flags:
`static`

Implements:
`public` *weighted_graph_protocol*
Extends:
`public` *graph_common*
Uses:
list
pairs

Remarks:
(none)

Inherited public predicates:
add_edge/5 add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 breadth_first_order/3 delete_edge/5 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edge/4 edges/2 empty/1 has_negative_cycle/1 has_path/3 is_bipartite/1 is_complete/1 is_sparse/1 max_path/5 min_distances/3 min_path/5 min_path_bellman_ford/5 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 path/3 reachable/3 vertices/2 wneighbors/3

- *Public predicates*
- *Protected predicates*
  - `winsert_neighbor/4`
  - `wremove_neighbor/4`
  - `wfind/3`
  - `wremove_vertex_from_all/3`
  - `relax_neighbors/7`
  - `pq_insert/3`
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

winsert_neighbor/4

Inserts a weighted neighbor into a sorted weighted neighbor list, replacing any existing entry for the same vertex.

Compilation flags:
`static`

Template:
`winsert_neighbor(WNeighbors,Vertex,Weight,NewWNeighbors)`
Mode and number of proofs:
`winsert_neighbor(+list,+vertex,+number,-list) - one`

wremove_neighbor/4

Removes a vertex from a sorted weighted neighbor list, unifying the weight. Fails if the vertex is not found.

Compilation flags:
`static`

Template:
`wremove_neighbor(WNeighbors,Vertex,Weight,NewWNeighbors)`
Mode and number of proofs:
`wremove_neighbor(+list,+vertex,-number,-list)` - `zero_or_one`

---

wfind/3

Finds the weight associated with a vertex in a weighted neighbor list.

Compilation flags:
`static`

Template:
`wfind(WNeighbors,Vertex,Weight)`
Mode and number of proofs:
`wfind(+list,+vertex,-number)` - `zero_or_one`

---

wremove_vertex_from_all/3

Removes a vertex from all weighted neighbor lists in a list of vertex-neighbors pairs.

Compilation flags:
`static`

Template:
`wremove_vertex_from_all(Pairs,Vertex,NewPairs)`
Mode and number of proofs:
`wremove_vertex_from_all(+list(pair),+vertex,-list(pair))` - `one`

---

relax_neighbors/7

Relaxes neighbors during Dijkstra shortest path computation, updating distances and priority queue.

Compilation flags:
`static`

Template:
`relax_neighbors(WNeighbors,Vertex,Distance,Queue,Dist,NewQueue,NewDist)`
Mode and number of proofs:
`relax_neighbors(+list,+vertex,+number,+list,+dictionary,-list,-dictionary)` - `one`

---

pq_insert/3

Inserts an element into a sorted priority queue (list of Distance-Vertex pairs).

Compilation flags:
`static`

Template:
`pq_insert(Queue,Item,NewQueue)`
Mode and number of proofs:
`pq_insert(+list,+pair,-list)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.108.16 weighted_graph_protocol

Protocol for weighted graph predicates, extending the common graph protocol with weighted edge operations.

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-25

Compilation flags:
`static`

Extends:

`public` *graph_protocol*

Remarks:

(none)

Inherited public predicates:

add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 breadth_first_order/3 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edges/2 empty/1 has_path/3 is_bipartite/1 is_complete/1 is_sparse/1 max_path/5 min_distances/3 min_path/5 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 path/3 reachable/3 vertices/2

- *Public predicates*
  - *edge/4*
  - *add_edge/5*
  - *delete_edge/5*
  - *wneighbors/3*
  - *min_path_bellman_ford/5*
  - *has_negative_cycle/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

edge/4

True iff there is an edge between `Vertex1` and `Vertex2` with weight `Weight` in `Graph`.

Compilation flags:

`static`

Template:

`edge(Vertex1,Vertex2,Weight,Graph)`

Mode and number of proofs:

`edge(+vertex,+vertex,?number,+graph)` - `zero_or_one`

add_edge/5

Adds a weighted edge between `Vertex1` and `Vertex2` with weight `Weight`.

Compilation flags:

```
static
```

Template:

```
add_edge(Graph,Vertex1,Vertex2,Weight,NewGraph)
```

Mode and number of proofs:

```
add_edge(+graph,+vertex,+vertex,+number,-graph) - one
```

---

delete_edge/5

Deletes the weighted edge between `Vertex1` and `Vertex2`. Unifies `Weight` with the weight of the deleted edge. The graph is unchanged if the edge does not exist.

Compilation flags:

```
static
```

Template:

```
delete_edge(Graph,Vertex1,Vertex2,Weight,NewGraph)
```

Mode and number of proofs:

```
delete_edge(+graph,+vertex,+vertex,?number,-graph) - one
```

---

wneighbors/3

Unifies `WNeighbors` with a list of `NeighborVertex-Weight` pairs for the neighbors of `Vertex`. Fails if `Vertex` is not in the graph.

Compilation flags:

```
static
```

Template:

```
wneighbors(Vertex,Graph,WNeighbors)
```

Mode and number of proofs:

```
wneighbors(+vertex,+graph,-list(pair)) - zero_or_one
```

---

min_path_bellman_ford/5

Finds the minimum cost path from `Vertex1` to `Vertex2` using the Bellman-Ford algorithm. Supports negative edge weights and fails if no path exists or if a reachable negative cycle makes the optimum undefined.

Compilation flags:
`static`

Template:
`min_path_bellman_ford(Vertex1,Vertex2,Graph,Path,Cost)`
Mode and number of proofs:
`min_path_bellman_ford(+vertex,+vertex,+graph,-list(vertex),-number)` - `zero_or_one`

---

has_negative_cycle/1

True iff `Graph` contains a negative-weight cycle.

Compilation flags:
`static`

Template:
`has_negative_cycle(Graph)`
Mode and number of proofs:
`has_negative_cycle(+graph)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.108.17 weighted_undirected_graph

Weighted undirected graph predicates using the AVL tree dictionary representation. Each edge is stored in both directions. Edge weights use a pair representation (Vertex-Weight in neighbor lists, (V1-V2)-Weight for edge lists).

Availability:
`logtalk_load(graphs(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-02-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *weighted_undirected_graph(avltree)*

Remarks:
(none)

Inherited public predicates:
add_edge/5 add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 articulation_points/2 breadth_first_order/3 bridges/2 connected_components/2 cycle/2 degree/3 delete_edge/5 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edge/4 edges/2 empty/1 graph_coloring/3 has_cycle/1 has_negative_cycle/1 has_path/3 is_bipartite/1 is_complete/1 is_connected/1 is_sparse/1 is_tree/1 max_path/5 max_tree/3 maximal_cliques/2 maximum_cliques/2 min_distances/3 min_path/5 min_path_bellman_ford/5 min_predecessors/3 min_tree/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 path/3 reachable/3 vertices/2 wneighbors/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.108.18 `weighted_undirected_graph(Dictionary)`

Weighted undirected graph predicates using a dictionary representation. Each edge is stored in both directions. Edge weights use a pair representation (`Vertex-Weight` in neighbor lists, `(Vertex1-Vertex2)-Weight` for edge lists). The parametric object parameter is the dictionary to use for the graph representation.

Availability:

`logtalk_load(graphs(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-09

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *weighted_graph_common(Dictionary)*
`public` *undirected_graph_common*

Uses:

list

Remarks:

(none)

Inherited public predicates:

add_edge/5 add_edges/3 add_vertex/3 add_vertices/3 all_pairs_min_paths/2 all_pairs_min_predecessors/2 articulation_points/2 breadth_first_order/3 bridges/2 cycle/2 delete_edge/5 delete_edges/3 delete_vertex/3 delete_vertices/3 depth_first_order/3 edge/4 edges/2 empty/1 graph_coloring/3 has_cycle/1 has_negative_cycle/1 has_path/3 is_bipartite/1 is_complete/1 is_connected/1 is_sparse/1 is_tree/1 max_path/5 maximal_cliques/2 maximum_cliques/2 min_distances/3 min_path/5 min_path_bellman_ford/5 min_predecessors/3 neighbors/3 new/1 new/2 new/3 number_of_edges/2 number_of_vertices/2 path/3 reachable/3 vertices/2 wneighbors/3

- *Public predicates*
  - *degree/3*
  - *connected_components/2*
  - *min_tree/3*
  - *max_tree/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

degree/3

Returns the degree (number of edges incident to the vertex) of `Vertex` in `Graph`.

Compilation flags:
`static`

Template:
`degree(Vertex,Graph,Degree)`
Mode and number of proofs:
`degree(+vertex,+graph,-integer)` - `zero_or_one`

---

connected_components/2

Returns the list of connected components (each a list of vertices).

Compilation flags:
`static`

Template:
`connected_components(Graph,Components)`
Mode and number of proofs:
`connected_components(+graph,-list(list))` - `one`

---

min_tree/3

Constructs a minimum spanning tree and returns its total weight.

Compilation flags:
`static`

Template:
`min_tree(Graph,Tree,Cost)`
Mode and number of proofs:
`min_tree(+graph,-graph,-number)` - `zero_or_one`

`max_tree/3`

Constructs a maximum spanning tree and returns its total weight.

Compilation flags:
`static`

Template:
`max_tree(Graph,Tree,Cost)`
Mode and number of proofs:
`max_tree(+graph,-graph,-number)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.109 gravatar

object

1.109.1 `gravatar`

Portable Gravatar profile client using the Gravatar REST API.

Availability:
`logtalk_load(gravatar(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
gravatar_api
http_client
http_core

list
os
sha256
url(Representation)

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *profile/2*
  - *profile/3*
  - *profile_response/3*
  - *email_hash/2*
  - *field/3*
  - *hash/2*
  - *display_name/2*
  - *profile_url/2*
  - *avatar_url/2*
  - *avatar_alt_text/2*
  - *location/2*
  - *job_title/2*
  - *company/2*
  - *description/2*
  - *pronouns/2*
  - *verified_accounts/2*
  - *section_visibility/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

profile/2

Retrieves a Gravatar profile JSON object for an email address using default options.

Compilation flags:
`static`

Template:
`profile(Email,Profile)`
Mode and number of proofs:
`profile(+atom,-term)` - `one_or_error`

Exceptions:

`Email` is a variable:
`instantiation_error`
`Email` is neither a variable nor an atom:
`type_error(atom,Email)`
The Gravatar API returns a non-success response:
`gravatar_api_error(Status,Body)`

`profile/3`

Retrieves a Gravatar profile JSON object for an email address using the given options.

Compilation flags:
`static`

Template:
`profile(Email,Profile,Options)`
Mode and number of proofs:
`profile(+atom,-term,+list(compound))` - `one_or_error`

Exceptions:
`Email` is a variable:
`instantiation_error`
`Email` is neither a variable nor an atom:
`type_error(atom,Email)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The Gravatar API returns a non-success response:
`gravatar_api_error(Status,Body)`

Remarks:

- Option `api_key(APIKey)`: Uses the given Gravatar API key as a Bearer token. When absent, the `LOGTALK_GRAVATAR_API_KEY` environment variable is used when defined.
- Option `base_url(URL)`: Overrides the Gravatar API base URL. The default is `https://api.gravatar.com/v3`.
- HTTP options: Options `headers/1`, `version/1`, `properties/1`, and `connection_options/1` are forwarded to the HTTP client.

---

profile_response/3

Retrieves the raw normalized HTTP response for a Gravatar profile request.

Compilation flags:
`static`

Template:
`profile_response(Email,Response,Options)`
Mode and number of proofs:
`profile_response(+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Email` is a variable:
`instantiation_error`
`Email` is neither a variable nor an atom:
`type_error(atom,Email)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

email_hash/2

Computes the Gravatar SHA-256 email hash after trimming leading and trailing whitespace and lowercasing ASCII letters.

Compilation flags:
`static`

Template:
`email_hash(Email,Hash)`
Mode and number of proofs:
`email_hash(+atom,-atom)` - `one_or_error`

Exceptions:
`Email` is a variable:
`instantiation_error`
`Email` is neither a variable nor an atom:
`type_error(atom,Email)`

---

field/3

Returns a top-level field from a decoded Gravatar profile JSON object.

Compilation flags:
`static`

Template:
`field(Profile,Field,Value)`
Mode and number of proofs:
`field(+term,+atom,?term)` - `zero_or_one`

---

hash/2

Returns the Gravatar profile hash field.

Compilation flags:
`static`

Template:
`hash(Profile,Value)`
Mode and number of proofs:
`hash(+term,?term)` - `zero_or_one`

---

display_name/2

Returns the Gravatar profile display name field.

Compilation flags:
`static`

Template:
`display_name(Profile,Value)`
Mode and number of proofs:
`display_name(+term,?term)` - `zero_or_one`

---

profile_url/2

Returns the Gravatar profile URL field.

Compilation flags:
```
static
```

Template:
```
profile_url(Profile,Value)
```
Mode and number of proofs:
```
profile_url(+term,?term) - zero_or_one
```

---

avatar_url/2

Returns the Gravatar avatar URL field.

Compilation flags:
```
static
```

Template:
```
avatar_url(Profile,Value)
```
Mode and number of proofs:
```
avatar_url(+term,?term) - zero_or_one
```

---

avatar_alt_text/2

Returns the Gravatar avatar alternative text field.

Compilation flags:
```
static
```

Template:
```
avatar_alt_text(Profile,Value)
```
Mode and number of proofs:
```
avatar_alt_text(+term,?term) - zero_or_one
```

---

location/2

Returns the Gravatar profile location field.

Compilation flags:
`static`

Template:
`location(Profile,Value)`
Mode and number of proofs:
`location(+term,?term)` - `zero_or_one`

---

job_title/2

Returns the Gravatar profile job title field.

Compilation flags:
`static`

Template:
`job_title(Profile,Value)`
Mode and number of proofs:
`job_title(+term,?term)` - `zero_or_one`

---

company/2

Returns the Gravatar profile company field.

Compilation flags:
`static`

Template:
`company(Profile,Value)`
Mode and number of proofs:
`company(+term,?term)` - `zero_or_one`

---

description/2

Returns the Gravatar profile description field.

Compilation flags:
`static`

Template:
`description(Profile,Value)`
Mode and number of proofs:
`description(+term,?term)` - `zero_or_one`

---

pronouns/2

Returns the Gravatar profile pronouns field.

Compilation flags:
`static`

Template:
`pronouns(Profile,Value)`
Mode and number of proofs:
`pronouns(+term,?term)` - `zero_or_one`

---

verified_accounts/2

Returns the Gravatar profile verified accounts field.

Compilation flags:
`static`

Template:
`verified_accounts(Profile,Value)`
Mode and number of proofs:
`verified_accounts(+term,?term)` - `zero_or_one`

`section_visibility/2`

Returns the Gravatar profile section visibility field.

Compilation flags:
`static`

Template:
`section_visibility(Profile,Value)`
Mode and number of proofs:
`section_visibility(+term,?term)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.109.2 `gravatar_api`

REST metadata and path construction helpers for the Gravatar public API.

Availability:
`logtalk_load(gravatar(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *rest*

Remarks:
(none)

Inherited public predicates:
api_info/1 *binary_body/2* *created_response/4* *form_body/2* *handle/2* *json_array_body/2* *json_body/2* *json_object_body/2* *json_response/4* *json_response/5* *no_content_response/2* *operations/1* *path_parameter/3* *problem_response/6* *query_parameter/3* *request_body/2* *request_header/3* *schema/2* *security/1* *security_scheme/2* *servers/1* *text_body/2*

- *Public predicates*
  - *profile_path/2*
- *Protected predicates*
  - *endpoint/5*
  - *profile/2*
- *Private predicates*
- *Operators*

Public predicates

profile_path/2

Builds the profile endpoint path for a SHA-256 email hash or profile URL slug.

Compilation flags:
`static`

Template:
`profile_path(ProfileIdentifier,Path)`
Mode and number of proofs:
`profile_path(+atom,-atom)` - `one`

Protected predicates

endpoint/5

REST endpoint descriptor for the Gravatar profile lookup operation.

Compilation flags:
`static`

Template:
`endpoint(Id,Method,Path,Action,Options)`
Mode and number of proofs:
`endpoint(?atom,?atom,?atom,?atom,?list(compound))` - `zero_or_more`

---

profile/2

Placeholder action for the remote Gravatar profile endpoint descriptor.

Compilation flags:
`static`

Template:
`profile(Request,Result)`
Mode and number of proofs:
`profile(+compound,--term)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.110 gsp_pattern_miner

object

1.110.1 gsp_pattern_miner

GSP sequential pattern miner for sequence datasets using level-wise candidate generation and pruning.

Availability:
`logtalk_load(gsp_pattern_miner(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *sequential_pattern_mining_common*
Uses:
avltree
format
list

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_pattern_miner/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *mine/2* *mine/3* *option/2* *option/3* *pattern_miner_options/2* *print_pattern_miner/1* *valid_option/1* *valid_options/1* *valid_pattern_miner/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pattern_miner_protocol, *sequence_dataset_protocol*, *prefix_span_pattern_miner*, *spade_pattern_miner*, *clo_span_pattern_miner*

1.111 hashes

category

1.111.1 `blake2_core(Key,DigestSize)`

- `Key` - A list of key bytes. Use the empty list for unkeyed hashing.
- `DigestSize` - Number of digest bytes to generate.

Common BLAKE2 hash function core.

Availability:

```
logtalk_load(hashes(loader))
```

Author: Paulo Moura
Version: 1:2:0
Date: 2026-10-12

Compilation flags:

```
static
```

Implements:
`public` *hash_digest_protocol*
`public` *hash_state_protocol*
Uses:
hash_common_32
list

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
  - *blake2_max_digest_size/1*
  - *blake2_max_key_size/1*
  - *blake2_block_size/1*
  - *blake2_round_count/1*
  - *blake2_initial_state/2*
  - *blake2_working_vector/4*
  - *blake2_block_words/2*
  - *blake2_g/10*
  - *blake2_finalize/4*
  - *blake2_state_bytes/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

blake2_max_digest_size/1

Returns the maximum digest size in bytes for the BLAKE2 variant.

Compilation flags:
`static`

Template:
`blake2_max_digest_size(MaxDigestSize)`
Mode and number of proofs:
`blake2_max_digest_size(--integer)` - `one`

blake2_max_key_size/1

Returns the maximum key size in bytes for the BLAKE2 variant.

Compilation flags:
`static`

Template:
`blake2_max_key_size(MaxKeySize)`
Mode and number of proofs:
`blake2_max_key_size(--integer)` - `one`

---

blake2_block_size/1

Returns the block size in bytes for the BLAKE2 variant.

Compilation flags:
`static`

Template:
`blake2_block_size(BlockSize)`
Mode and number of proofs:
`blake2_block_size(--integer)` - `one`

---

blake2_round_count/1

Returns the number of compression rounds for the BLAKE2 variant.

Compilation flags:
`static`

Template:
`blake2_round_count(RoundCount)`
Mode and number of proofs:
`blake2_round_count(--integer)` - `one`

---

blake2_initial_state/2

Computes the initial hash state from the parameter word.

Compilation flags:
```
static
```

Template:
```
blake2_initial_state(ParameterWord,InitialState)
```
Mode and number of proofs:
```
blake2_initial_state(+integer,--list(integer)) - one
```

---

blake2_working_vector/4

Initializes the compression working vector from the hash state, byte counter, and final block flag.

Compilation flags:
```
static
```

Template:
```
blake2_working_vector(State,Total,Final,WorkingVector)
```
Mode and number of proofs:
```
blake2_working_vector(+list(integer),+integer,+boolean,--list(integer)) - one
```

---

blake2_block_words/2

Decodes a block of bytes into the message words used by the compression function.

Compilation flags:
```
static
```

Template:
```
blake2_block_words(Block,MessageWords)
```
Mode and number of proofs:
```
blake2_block_words(+list(byte),--list(integer)) - one
```

---

blake2_g/10

Applies the BLAKE2 G mixing function to four scalar working words and two message words.

Compilation flags:
`static`

Template:
`blake2_g(A0,B0,C0,D0,X,Y,A,B,C,D)`
Mode and number of proofs:
`blake2_g(+integer,+integer,+integer,+integer,+integer,+integer,--integer,--integer,--integer,--integer)` - `one`

---

blake2_finalize/4

Finalizes the hash state by combining it with the compression working vector.

Compilation flags:
`static`

Template:
`blake2_finalize(HashState,WorkingVector,Index,FinalState)`
Mode and number of proofs:
`blake2_finalize(+list(integer),+list(integer),+integer,--list(integer))` - `one`

---

blake2_state_bytes/2

Encodes the hash state words as little-endian bytes.

Compilation flags:
`static`

Template:
`blake2_state_bytes(State,Bytes)`
Mode and number of proofs:
`blake2_state_bytes(+list(integer),--list(byte))` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.111.2 `blake2b`

BLAKE2b hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-30

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *blake2b([],64)*

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

blake2b(Key,DigestSize), *blake2s*, *sha512*, *sha512_256*

object

1.111.3 `blake2b(Key,DigestSize)`

- `Key` - A list of 0 to 64 key bytes. Use the empty list for unkeyed hashing.
- `DigestSize` - Number of digest bytes to generate, from 1 to 64.

BLAKE2b hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *blake2_core(Key,DigestSize)*

Uses:
byte_order
hash_common_64
list

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

blake2b, *blake2s(Key,DigestSize)*, *sha512*, *sha512_256*

object

1.111.4 `blake2s`

BLAKE2s hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-30

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *blake2s([],32)*

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

blake2s(Key,DigestSize), *blake2b*, *md5*, *sha256*

object

1.111.5 `blake2s(Key,DigestSize)`

- `Key` - A list of 0 to 32 key bytes. Use the empty list for unkeyed hashing.
- `DigestSize` - Number of digest bytes to generate, from 1 to 32.

BLAKE2s hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *blake2_core(Key,DigestSize)*
Uses:
byte_order
hash_common_32
list

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

blake2s, *blake2b(Key,DigestSize)*, *md5*, *sha256*

object

1.111.6 `crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength)`

- `Polynomial` - Canonical non-reflected CRC-32 polynomial.
- `Initial` - Initial CRC accumulator value.
- `FinalXor` - Final xor value.
- `AppendLength` - Boolean flag controlling whether the message length is appended as little-endian bytes.

Parametric non-reflected CRC-32 hash function using a canonical polynomial, configurable initial value and final xor value, and optional appended little-endian length bytes.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-10-12

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public hash_state_protocol
```

Uses:

hash_common_32
list

Remarks:

(none)

Inherited public predicates:

final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

crc32_reflected(Polynomial), *crc32b*, *crc32c*, *crc32posix*, *crc32mpeg2*, *crc32bzip2*, *crc32q*, *murmurhash3_x86_32*, *fnv1a_32*

object

1.111.7 `crc32_reflected(Polynomial)`

- `Polynomial` - Reflected CRC-32 polynomial.

Parametric reflected CRC-32 hash function using initial value 0xFFFFFFFF and final xor 0xFFFFFFFF.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_state_protocol*
Uses:
hash_common_32

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength), *crc32b*, *crc32c*, *crc32posix*, *crc32mpeg2*, *crc32bzip2*, *crc32q*, *murmurhash3_x86_32*, *fnv1a_32*

object

1.111.8 `crc32b`

CRC-32/ISO-HDLC hash function using the reflected polynomial 0xEDB88320, initial value 0xFFFFFFFF, and final xor 0xFFFFFFFF.

Availability:

```
logtalk_load(hashes(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public crc32_reflected(3988292384)
```

Remarks:

(none)

Inherited public predicates:

final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

crc32_reflected(Polynomial), *crc32c*, *crc32posix*, *crc32mpeg2*, *crc32bzip2*, *crc32q*, *murmurhash3_x86_32*, *fnv1a_32*

object

1.111.9 `crc32bzip2`

CRC-32/BZIP2 hash function using the canonical polynomial 0x04C11DB7, initial value 0xFFFFFFFF, and final xor 0xFFFFFFFF.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *crc32_non_reflected(79764919,4294967295,4294967295,false)*

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength), *crc32mpeg2*, *crc32posix*, *crc32q*, *crc32_reflected(Polynomial)*, *crc32b*, *crc32c*, *murmurhash3_x86_32*, *fnv1a_32*

object

1.111.10 `crc32c`

CRC-32C/Castagnoli hash function using the reflected polynomial 0x82F63B78, initial value 0xFFFFFFFF, and final xor 0xFFFFFFFF.

Availability:

```
logtalk_load(hashes(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public crc32_reflected(2197175160)
```

Remarks:

(none)

Inherited public predicates:

final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

crc32_reflected(Polynomial), *crc32b*, *crc32posix*, *crc32mpeg2*, *crc32bzip2*, *crc32q*, *murmurhash3_x86_32*, *fnv1a_32*

object

1.111.11 `crc32mpeg2`

CRC-32/MPEG-2 hash function using the canonical polynomial 0x04C11DB7, initial value 0xFFFFFFFF, and final xor 0x00000000.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *crc32_non_reflected(79764919,4294967295,0,false)*

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength), *crc32posix*, *crc32bzip2*, *crc32q*, *crc32_reflected(Polynomial)*, *crc32b*, *crc32c*, *murmurhash3_x86_32*, *fnv1a_32*

object

1.111.12 `crc32posix`

CRC-32/POSIX (cksum) hash function using the canonical polynomial 0x04C11DB7, initial value 0x00000000, appended little-endian length bytes, and final xor 0xFFFFFFFF.

Availability:

```
logtalk_load(hashes(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public crc32_non_reflected(79764919,0,4294967295,true)
```

Remarks:

(none)

Inherited public predicates:

final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength), *crc32_reflected(Polynomial)*, *crc32b*, *crc32c*, *crc32mpeg2*, *crc32bzip2*, *crc32q*, *murmurhash3_x86_32*, *fnv1a_32*

object

1.111.13 `crc32q`

CRC-32Q hash function, also used by AIXM-style formats, using the canonical polynomial 0x814141AB, initial value 0x00000000, and final xor 0x00000000.

Availability:

```
logtalk_load(hashes(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-05

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public
```
crc32_non_reflected(2168537515,0,0,false)

Remarks:

(none)

Inherited public predicates:

final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength), *crc32mpeg2*, *crc32bzip2*, *crc32posix*, *crc32_reflected(Polynomial)*, *crc32b*, *crc32c*, *murmurhash3_x86_32*, *fnv1a_32*

object

1.111.14 djb2_32

DJB2 32-bit hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_state_protocol*
Uses:
hash_common_32

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

djb2_64, *sdbm_32*, *fnv1a_32*

object

1.111.15 djb2_64

DJB2 64-bit hash function.

Availability:
```
logtalk_load(hashes(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-16

Compilation flags:
```
static, context_switching_calls
```

Implements:
```
public hash_state_protocol
```
Uses:
hash_common_64

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 hash/2 new_hash_state/1 update_hash_state/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

djb2_32, *sdbm_64*, *fnv1a_64*

object

1.111.16 `fips202_fixed_hash(RateBytes,Suffix,OutputBytes)`

Common implementation of the fixed-size FIPS 202 SHA-3 variants suitable for use with HMAC.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-04-15

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_digest_protocol*
Extends:
`public` *fips202_hash(RateBytes,Suffix,OutputBytes)*

Remarks:
(none)

Inherited public predicates:

block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.111.17 `fips202_hash(RateBytes,Suffix,OutputBytes)`

Common implementation of the standardized FIPS 202 SHA-3 and SHAKE variants using the Keccak-f[1600] permutation.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_state_protocol*
Uses:
byte_order
hash_common_32
hash_common_64
list

Remarks:
(none)

Inherited public predicates:

final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
  - *absorb/2*
  - *squeeze/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

absorb/2

Absorbs the input bytes into the Keccak state and returns the resulting state after the final padded block permutation.

Compilation flags:

```
static
```

Template:

```
absorb(Bytes,State)
```

Mode and number of proofs:

```
absorb(+list(byte),--list(integer)) - one
```

---

squeeze/3

Squeezes the requested number of output bytes from a Keccak state.

Compilation flags:

```
static
```

Template:

```
squeeze(State,OutputBytes,DigestBytes)
```

Mode and number of proofs:

```
squeeze(+list(integer),+integer,--list(byte)) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.111.18 `fnv1a_32`

FNV-1a 32-bit hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_state_protocol*
Uses:
hash_common_32

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

fnv1a_64, *djb2_32*, *sdbm_32*

object

1.111.19 `fnv1a_64`

FNV-1a 64-bit hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_state_protocol*
Uses:
hash_common_64

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

fnv1a_32, *djb2_64*, *sdbm_64*

object

1.111.20 `hash_common_32`

Auxiliary predicates for the hashes library 32-bit algorithms.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2026-08-04

Compilation flags:
`static, context_switching_calls`

Uses:
byte_order
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *word32_hex/2*
  - *bytes_hex/2*
  - *mask32/1*

      - *add32/3*
      - *add32/4*
      - *add32/5*
      - *mul32/3*
      - *rol32/3*
      - *ror32/3*
      - *pad_md/4*
      - *pad_md_tail/5*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

word32_hex/2

Converts a 32-bit word into an 8-digit lowercase hexadecimal atom.

Compilation flags:
```
static
```

Template:
```
word32_hex(Word,Hex)
```
Mode and number of proofs:
```
word32_hex(+integer,-atom) - one
```

---

bytes_hex/2

Converts a list of bytes into a lowercase hexadecimal atom.

Compilation flags:
```
static
```

Template:
```
bytes_hex(Bytes,Hex)
```
Mode and number of proofs:
```
bytes_hex(+list(integer),-atom) - one
```

---

mask32/1

Returns the 32-bit mask value.

Compilation flags:
`static`

Template:
`mask32(Mask)`
Mode and number of proofs:
`mask32(-integer)` - `one`

---

add32/3

Adds two integers modulo 2^32.

Compilation flags:
`static`

Template:
`add32(A,B,Sum)`
Mode and number of proofs:
`add32(+integer,+integer,-integer)` - `one`

---

add32/4

Adds three integers modulo 2^32.

Compilation flags:
`static`

Template:
`add32(A,B,C,Sum)`
Mode and number of proofs:
`add32(+integer,+integer,+integer,-integer)` - `one`

---

add32/5

Adds four integers modulo 2^32.

Compilation flags:
```
static
```

Template:
```
add32(A,B,C,D,Sum)
```
Mode and number of proofs:
```
add32(+integer,+integer,+integer,+integer,-integer) - one
```

---

mul32/3

Multiplies two integers modulo 2^32. Computed via 16-bit limb decomposition so that no intermediate result exceeds about 2**49, avoiding integer overflow errors on backends with bounded integer arithmetic.

Compilation flags:
```
static
```

Template:
```
mul32(A,B,Product)
```
Mode and number of proofs:
```
mul32(+integer,+integer,-integer) - one
```

---

rol32/3

Rotates a 32-bit word left by the given number of bits.

Compilation flags:
```
static
```

Template:
```
rol32(Value,Shift,Rotated)
```
Mode and number of proofs:
```
rol32(+integer,+integer,-integer) - one
```

---

ror32/3

Rotates a 32-bit word right by the given number of bits.

Compilation flags:
```
static
```

Template:
```
ror32(Value,Shift,Rotated)
```
Mode and number of proofs:
```
ror32(+integer,+integer,-integer) - one
```

---

pad_md/4

Pads a message using MD-style padding with a little-endian or big-endian length field.

Compilation flags:
```
static
```

Template:
```
pad_md(Endian,Bytes,LengthFieldBytes,PaddedBytes)
```
Mode and number of proofs:
```
pad_md(+little_big,+list(integer),+integer,-list(integer)) - one
```

---

pad_md_tail/5

Pads the final, less-than-one-block tail of a message using MD-style padding given the total message length in bytes. For use when the message has already been consumed block by block during segmented/incremental hashing, so that only the unconsumed tail, and not the whole message, needs to be available.

Compilation flags:
```
static
```

Template:
```
pad_md_tail(Endian,TailBytes,TotalLength,LengthFieldBytes,PaddedBytes)
```
Mode and number of proofs:
```
pad_md_tail(+little_big,+list(integer),+integer,+integer,-list(integer)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.111.21 hash_common_64

Auxiliary predicates for the hashes library 64-bit algorithms.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-08-04

Compilation flags:
`static, context_switching_calls`

Uses:
byte_order
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *word64_hex/2*
  - *mask64/1*
  - *add64/3*
  - *mul64/3*
  - *rol64/3*
  - *xor64/3*
  - *or64/3*
  - *and64/3*
  - *not64/2*
  - *shl64/3*
  - *shr64/3*

    - *pad_md_tail/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

word64_hex/2

Converts a 64-bit word into a 16-digit lowercase hexadecimal atom.

Compilation flags:
`static`

Template:
`word64_hex(Word,Hex)`
Mode and number of proofs:
`word64_hex(+integer,-atom)` - `one`

---

mask64/1

Returns the 64-bit mask value.

Compilation flags:
`static`

Template:
`mask64(Mask)`
Mode and number of proofs:
`mask64(-integer)` - `one`

---

add64/3

Adds two integers modulo 2^64.

Compilation flags:
`static`

Template:
`add64(A,B,Sum)`
Mode and number of proofs:
`add64(+integer,+integer,-integer)` - `one`

---

mul64/3

Multiplies two integers modulo 2^64.

Compilation flags:
```
static
```

Template:
```
mul64(A,B,Product)
```
Mode and number of proofs:
```
mul64(+integer,+integer,-integer) - one
```

---

rol64/3

Rotates a 64-bit word left by the given number of bits.

Compilation flags:
```
static
```

Template:
```
rol64(Value,Shift,Rotated)
```
Mode and number of proofs:
```
rol64(+integer,+integer,-integer) - one
```

---

xor64/3

Computes the bitwise exclusive-or of two integers modulo 2^64.

Compilation flags:
```
static
```

Template:
```
xor64(A,B,Xor)
```
Mode and number of proofs:
```
xor64(+integer,+integer,-integer) - one
```

---

or64/3

Computes the bitwise disjunction of two integers modulo 2^64.

Compilation flags:
```
static
```

Template:
```
or64(A,B,Or)
```
Mode and number of proofs:
```
or64(+integer,+integer,-integer) - one
```

---

and64/3

Computes the bitwise conjunction of two integers modulo 2^64.

Compilation flags:
```
static
```

Template:
```
and64(A,B,And)
```
Mode and number of proofs:
```
and64(+integer,+integer,-integer) - one
```

---

not64/2

Computes the bitwise complement of an integer modulo 2^64.

Compilation flags:
```
static
```

Template:
```
not64(Value,Complement)
```
Mode and number of proofs:
```
not64(+integer,-integer) - one
```

---

shl64/3

Shifts a 64-bit word left by the given number of bits and masks the result.

Compilation flags:
```
static
```

Template:
```
shl64(Value,Shift,Shifted)
```
Mode and number of proofs:
```
shl64(+integer,+integer,-integer) - one
```

---

shr64/3

Shifts a 64-bit word right by the given number of bits after masking the input.

Compilation flags:
```
static
```

Template:
```
shr64(Value,Shift,Shifted)
```
Mode and number of proofs:
```
shr64(+integer,+integer,-integer) - one
```

---

pad_md_tail/3

Pads the final, less-than-one-block tail of a message using SHA-512-style MD padding (128-byte blocks, a big-endian 128-bit length field) given the total message length in bytes. For use when the message has already been consumed block by block during segmented/incremental hashing, so that only the unconsumed tail, and not the whole message, needs to be available.

Compilation flags:
```
static
```

Template:
```
pad_md_tail(TailBytes,TotalLength,PaddedBytes)
```
Mode and number of proofs:
```
pad_md_tail(+list(integer),+integer,-list(integer)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.111.22 `hash_digest_protocol`

Protocol for fixed-size cryptographic hash functions exposing raw digests and digest metadata.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-12

Compilation flags:
`static`

Extends:
`public` *hash_protocol*

Remarks:
(none)

Inherited public predicates:
hash/2

- *Public predicates*
  - *digest/2*
  - *digest_size/1*
  - *block_size/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

digest/2

Computes the digest for a list of bytes and returns it as a list of bytes.

Compilation flags:

```
static
```

Template:

```
digest(Bytes,Digest)
```

Mode and number of proofs:

```
digest(+list(byte),--list(byte)) - one_or_error
```

Exceptions:

The key or digest size is not sufficiently instantiated::

```
instantiation_error
```

The digest size is invalid for BLAKE2::

```
domain_error(blake2_digest_size,DigestSize)
```

The key is ground but invalid for BLAKE2::

```
domain_error(blake2_key,Key)
```

---

digest_size/1

Returns the digest size in bytes.

Compilation flags:

```
static
```

Template:

```
digest_size(DigestSize)
```

Mode and number of proofs:

```
digest_size(--integer) - one_or_error
```

Exceptions:

The key or digest size is not sufficiently instantiated::

```
instantiation_error
```

The digest size is invalid for BLAKE2::

```
domain_error(blake2_digest_size,DigestSize)
```

The key is ground but invalid for BLAKE2::

```
domain_error(blake2_key,Key)
```

---

block_size/1

Returns the hash block size in bytes.

Compilation flags:
`static`

Template:
`block_size(BlockSize)`
Mode and number of proofs:
`block_size(--integer)` - `one_or_error`

Exceptions:
The key or digest size is not sufficiently instantiated::
`instantiation_error`
The digest size is invalid for BLAKE2::
`domain_error(blake2_digest_size,DigestSize)`
The key is ground but invalid for BLAKE2::
`domain_error(blake2_key,Key)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.111.23 hash_protocol

Hashing protocol. Hash values are returned as lowercase hexadecimal atoms using the output width of each algorithm or extensible-output function instance.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-12

Compilation flags:
`static`

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *hash/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

hash/2

Computes the hash for a list of bytes.

Compilation flags:
`static`

Template:
`hash(Bytes,Hash)`

Mode and number of proofs:
`hash(+list(byte),--atom)` - `one_or_error`

Exceptions:
The key, digest size, or output length is not sufficiently instantiated::
`instantiation_error`
The digest size is invalid for BLAKE2::
`domain_error(blake2_digest_size,DigestSize)`
The key is ground but invalid for BLAKE2::
`domain_error(blake2_key,Key)`
The key is ground but not a list of 16 bytes for SipHash::
`domain_error(siphash_key,Key)`
The output length is ground but not a nonnegative integer for SHAKE::
`domain_error(shake_output_size,OutputBytes)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.111.24 `hash_state_protocol`

Protocol for genuine incremental hashing via an opaque caller-driven state. Retained state is bounded independently of total message length. Leftover buffers are smaller than one block or rate, except BLAKE2 may retain one full final block. Processing may allocate temporary lists proportional to the current chunk. Callers must release processed chunks and old states to obtain bounded retained memory.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-12

Compilation flags:
`static`

Extends:
`public` *hash_protocol*

Remarks:
(none)

Inherited public predicates:
hash/2

- *Public predicates*
  - *new_hash_state/1*
  - *update_hash_state/3*
  - *final_hash_state/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new_hash_state/1

Creates a fresh hashing state, equivalent to having hashed the empty message so far. The state is an opaque, algorithm-specific term; callers should only pass it to update_hash_state/3 and final_hash_state/2, never inspect or construct it directly.

Compilation flags:

```
static
```

Template:

```
new_hash_state(State)
```

Mode and number of proofs:

```
new_hash_state(--nonvar) - one_or_error
```

Exceptions:

The key, digest size, or output length is not sufficiently instantiated::

```
instantiation_error
```

The digest size is invalid for BLAKE2::

```
domain_error(blake2_digest_size,DigestSize)
```

The key is ground but invalid for BLAKE2::

```
domain_error(blake2_key,Key)
```

The key is ground but not a list of 16 bytes for SipHash::

```
domain_error(siphash_key,Key)
```

The output length is ground but not a nonnegative integer for SHAKE::

```
domain_error(shake_output_size,OutputBytes)
```

---

update_hash_state/3

Folds one chunk of bytes into a hashing state, returning the updated state. Can be called repeatedly, once per chunk, to fold in an arbitrarily long message without ever holding more than one chunk and the (bounded-size) state in memory at the same time. The chunks do not need to be the same length, and the empty chunk is allowed and is a no-op.

Compilation flags:

```
static
```

Template:

```
update_hash_state(State,Bytes,NewState)
```

Mode and number of proofs:

```
update_hash_state(+nonvar,+list(byte),--nonvar) - one
```

---

final_hash_state/2

Finalizes a hashing state and returns the resulting hash, in the same representation as hash/2. The state must not be reused after finalization.

Compilation flags:
`static`

Template:
`final_hash_state(State,Hash)`
Mode and number of proofs:
`final_hash_state(+nonvar,--atom)` - `one_or_error`

Exceptions:
The key, digest size, or output length is not sufficiently instantiated::
`instantiation_error`
The digest size is invalid for BLAKE2::
`domain_error(blake2_digest_size,DigestSize)`
The key is ground but invalid for BLAKE2::
`domain_error(blake2_key,Key)`
The key is ground but not a list of 16 bytes for SipHash::
`domain_error(siphash_key,Key)`
The output length is ground but not a nonnegative integer for SHAKE::
`domain_error(shake_output_size,OutputBytes)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.111.25 md5

MD5 hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:5:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_digest_protocol*
`public` *hash_state_protocol*

Uses:
byte_order
hash_common_32
list

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sha1, *sha256*

object

1.111.26 murmurhash3_x64_128

MurmurHash3 x64 128-bit hash function with seed 0.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_state_protocol*
Uses:
hash_common_64
list

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

murmurhash3_x86_32, *murmurhash3_x86_128*

object

1.111.27 `murmurhash3_x86_128`

MurmurHash3 x86 128-bit hash function with seed 0.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-08-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_state_protocol*
Uses:
byte_order
hash_common_32
list

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

murmurhash3_x86_32, *murmurhash3_x64_128*

object

1.111.28 `murmurhash3_x86_32`

MurmurHash3 x86 32-bit hash function with seed 0.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_state_protocol*
Uses:
byte_order
hash_common_32
list

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

murmurhash3_x86_128, *murmurhash3_x64_128*

object

1.111.29 `sdbm_32`

sdbm 32-bit hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_state_protocol*
Uses:
hash_common_32

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sdbm_64, *djb2_32*, *fnv1a_32*

object

1.111.30 `sdbm_64`

sdbm 64-bit hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_state_protocol*
Uses:
hash_common_64

Remarks:
(none)

Inherited public predicates:

final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sdbm_32, *djb2_64*, *fnv1a_64*

object

1.111.31 `sha1`

SHA-1 hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:5:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_digest_protocol*
`public` *hash_state_protocol*
Uses:
byte_order
hash_common_32

list

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

md5, *sha256*, *sha224*

object

1.111.32 `sha224`

SHA-224 hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:

`public` *hash_digest_protocol*
`public` *hash_state_protocol*

Uses:

byte_order
hash_common_32
list

Remarks:

(none)

Inherited public predicates:

block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sha256, *sha1*, *sha384*

object

1.111.33 `sha256`

SHA-256 hash function.

Availability:

`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:5:0

Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_digest_protocol*
`public` *hash_state_protocol*

Uses:
byte_order
hash_common_32
list

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

md5, *sha1*, *sha224*

object

1.111.34 sha384

SHA-384 hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_digest_protocol*
`public` *hash_state_protocol*

Uses:
byte_order
hash_common_32
hash_common_64
list

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sha512, *sha512_256*, *sha256*, *sha224*

object

1.111.35 `sha3_224`

FIPS 202 SHA3-224 hash function.

Availability:

```
logtalk_load(hashes(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-04-15

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public fips202_fixed_hash(144,6,28)
```

Remarks:

(none)

Inherited public predicates:

block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sha3_256, *sha3_384*, *sha3_512*, *shake128(OutputBytes)*, *shake256(OutputBytes)*

object

1.111.36 `sha3_256`

FIPS 202 SHA3-256 hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-04-15

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *fips202_fixed_hash(136,6,32)*

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sha3_224, *sha3_384*, *sha3_512*, *sha256*, *shake128(OutputBytes)*, *shake256(OutputBytes)*

object

1.111.37 `sha3_384`

FIPS 202 SHA3-384 hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-04-15

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *fips202_fixed_hash(104,6,48)*

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sha3_224, *sha3_256*, *sha3_512*, *shake128(OutputBytes)*, *shake256(OutputBytes)*

object

1.111.38 `sha3_512`

FIPS 202 SHA3-512 hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-04-15

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *fips202_fixed_hash(72,6,64)*

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sha3_224, *sha3_256*, *sha3_384*, *shake128(OutputBytes)*, *shake256(OutputBytes)*

object

1.111.39 `sha512`

SHA-512 hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_digest_protocol*
`public` *hash_state_protocol*
Uses:
byte_order
hash_common_32
hash_common_64
list

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sha512_256, *sha384*, *sha256*, *sha1*

object

1.111.40 `sha512_256`

SHA-512/256 hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hash_digest_protocol*
`public` *hash_state_protocol*
Uses:
byte_order
hash_common_32
hash_common_64
list

Remarks:
(none)

Inherited public predicates:
block_size/1 *digest/2* *digest_size/1* *final_hash_state/2* *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sha256, *sha1*, *sha384*

object

1.111.41 `shake128(OutputBytes)`

- `OutputBytes` - Number of output bytes to generate.

FIPS 202 SHAKE128 extensible-output function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-04

Compilation flags:
`static, context_switching_calls`

Extends:

`public` *fips202_hash(168,31,OutputBytes)*

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

shake256(OutputBytes), *sha3_224*, *sha3_256*, *sha3_384*, *sha3_512*

object

1.111.42 `shake256(OutputBytes)`

- `OutputBytes` - Number of output bytes to generate.

FIPS 202 SHAKE256 extensible-output function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-04

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *fips202_hash(136,31,OutputBytes)*

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

shake128(OutputBytes), *sha3_224*, *sha3_256*, *sha3_384*, *sha3_512*

object

1.111.43 `siphash_2_4`

SipHash-2-4 hash function using the standard reference key 00 01 02 … 0f.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-04

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *siphash_2_4([0,1,2,3,4,5,6,7,8,9,10,11,12,13,14,15])*

Remarks:
(none)

Inherited public predicates:
final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

siphash_2_4(Key), *fnv1a_64*, *crc32b*, *crc32c*

object

1.111.44 `siphash_2_4(Key)`

- `Key` - A list of 16 bytes.

SipHash-2-4 keyed hash function.

Availability:
`logtalk_load(hashes(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-10-12

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public hash_state_protocol
```

Uses:

byte_order
hash_common_64
list

Remarks:

(none)

Inherited public predicates:

final_hash_state/2 *hash/2* *new_hash_state/1* *update_hash_state/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

siphash_2_4, *fnv1a_64*, *crc32b*, *crc32c*

1.112 hdbscan_clusterer

object

1.112.1 hdbscan_clusterer

HDBSCAN clusterer for continuous datasets using a mutual-reachability hierarchy, condensed tree pruning, and stability-based cluster selection.

Availability:

```
logtalk_load(hdbscan_clusterer(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public clusterer_common
```

Uses:

format
list
numberlist
type
union_find

Remarks:

(none)

Inherited public predicates:

check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *dbscan_clusterer*, *optics_clusterer*

1.113 heaps

object

1.113.1 `binary_heap(Order)`

- `Order` - Either `<` for a min heap or `>` for a max heap.

Heap implementation, parameterized by the order to be used to compare keys (`<` or `>`).

Availability:
`logtalk_load(heaps(loader))`

Author: Richard O'Keefe; adapted to Logtalk by Paulo Moura and Victor Lagerkvist.
Version: 1:2:0
Date: 2026-01-28

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *heap_protocol*
Extends:
`public` *compound*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_heap/2 as_list/2 check/1 delete/4 depth/2 empty/1 ground/1 insert/4 insert_all/3 merge/3 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 size/2 subsumes/2 subterm/2 top/3 top_next/5 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

binary_heap_min, *binary_heap_max*

object

1.113.2 `binary_heap_max`

Max-heap implementation. Uses standard order to compare keys.

Availability:
`logtalk_load(heaps(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2010-02-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *binary_heap(>)*

Remarks:
(none)

Inherited public predicates:

(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_heap/2 as_list/2 check/1 delete/4 depth/2 empty/1 ground/1 insert/4 insert_all/3 merge/3 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 size/2 subsumes/2 subterm/2 top/3 top_next/5 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

binary_heap_min, *binary_heap(Order)*

object

1.113.3 `binary_heap_min`

Min-heap implementation. Uses standard order to compare keys.

Availability:
`logtalk_load(heaps(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2010-02-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *binary_heap(<)*

Remarks:
(none)

Inherited public predicates:

(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_heap/2 as_list/2 check/1 delete/4 depth/2 empty/1 ground/1 insert/4 insert_all/3 merge/3 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 size/2 subsumes/2 subterm/2 top/3 top_next/5 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

binary_heap_max, *binary_heap(Order)*

object

1.113.4 `heap(Order)`

- `Order` - Either `<` for a min heap or `>` for a max heap.

Heap implementation, parameterized by the order to be used to compare keys (`<` or `>`). Deprecated. Use the `binary_heap/1` object instead.

Availability:

```
logtalk_load(heaps(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-01-28

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *binary_heap(Order)*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_heap/2 as_list/2 check/1 delete/4 depth/2 empty/1 ground/1 insert/4 insert_all/3 merge/3 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 size/2 subsumes/2 subterm/2 top/3 top_next/5 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

binary_heap(Order)

protocol

1.113.5 `heap_protocol`

Heap protocol. Key-value pairs are represented as `Key-Value`.

Availability:
`logtalk_load(heaps(loader))`

Author: Richard O'Keefe; adapted to Logtalk by Paulo Moura and Victor Lagerkvist.
Version: 1:0:1
Date: 2010-11-13

Compilation flags:
`static`

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *insert/4*
  - *insert_all/3*
  - *delete/4*
  - *merge/3*
  - *empty/1*
  - *size/2*
  - *as_list/2*
  - *as_heap/2*
  - *top/3*
  - *top_next/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

insert/4

Inserts the new pair into a heap, returning the updated heap.

Compilation flags:
`static`

Template:
`insert(Key,Value,Heap,NewHeap)`

Mode and number of proofs:
`insert(+key,+value,+heap,-heap)` - `one`

insert_all/3

Inserts a list of pairs into a heap, returning the updated heap.

Compilation flags:
```
static
```

Template:
```
insert_all(List,Heap,NewHeap)
```
Mode and number of proofs:
```
insert_all(@list(pairs),+heap,-heap) - one
```

---

delete/4

Deletes and returns the top pair in a heap returning the updated heap.

Compilation flags:
```
static
```

Template:
```
delete(Heap,TopKey,TopValue,NewHeap)
```
Mode and number of proofs:
```
delete(+heap,?key,?value,-heap) - zero_or_one
```

---

merge/3

Merges two heaps.

Compilation flags:
```
static
```

Template:
```
merge(Heap1,Heap2,NewHeap)
```
Mode and number of proofs:
```
merge(+heap,+heap,-heap) - one
```

---

empty/1

True if the heap is empty.

Compilation flags:
`static`

Template:
`empty(Heap)`
Mode and number of proofs:
`empty(@heap)` - `zero_or_one`

---

size/2

Returns the number of heap elements.

Compilation flags:
`static`

Template:
`size(Heap,Size)`
Mode and number of proofs:
`size(+heap,?integer)` - `zero_or_one`

---

as_list/2

Returns the current set of pairs in the heap as a list, sorted into ascending order of the keys.

Compilation flags:
`static`

Template:
`as_list(Heap,List)`
Mode and number of proofs:
`as_list(+heap,-list)` - `one`

---

as_heap/2

Constructs a heap from a list of pairs.

Compilation flags:
`static`

Template:
`as_heap(List,Heap)`
Mode and number of proofs:
`as_heap(+list,-heap)` - `one`

---

top/3

Returns the top pair in the heap. Fails if the heap is empty.

Compilation flags:
`static`

Template:
`top(Heap,TopKey,TopValue)`
Mode and number of proofs:
`top(+heap,?key,?value)` - `zero_or_one`

---

top_next/5

Returns the top pair and the next pair in the heap. Fails if the heap does not have at least two elements.

Compilation flags:
`static`

Template:
`top_next(Heap,TopKey,TopValue,NextKey,NextValue)`
Mode and number of proofs:
`top_next(+heap,?key,?value,?key,?value)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

heap(Order), *pairing_heap(Order)*

protocol

1.113.6 `heapp`

Heap protocol. Deprecated. Use the `heap_protocol` protocol instead.

Availability:
`logtalk_load(heaps(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-01-28

Compilation flags:
`static`

Extends:
`public` *heap_protocol*

Remarks:
(none)

Inherited public predicates:
as_heap/2 *as_list/2* *delete/4* *empty/1* *insert/4* *insert_all/3* *merge/3* *size/2* *top/3* *top_next/5*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

heap_protocol

object

1.113.7 `maxheap`

Max-heap implementation. Uses standard order to compare keys. Deprecated. Use the `binary_heap_max` object instead.

Availability:
`logtalk_load(heaps(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2010-02-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *binary_heap(>)*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_heap/2 as_list/2 check/1 delete/4 depth/2 empty/1 ground/1 insert/4 insert_all/3 merge/3 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 size/2 subsumes/2 subterm/2 top/3 top_next/5 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

binary_heap_max

object

1.113.8 `minheap`

Min-heap implementation. Uses standard order to compare keys. Deprecated. Use the `binary_heap_min` object instead.

Availability:
`logtalk_load(heaps(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2010-02-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *binary_heap(<)*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_heap/2 as_list/2 check/1 delete/4 depth/2 empty/1 ground/1 insert/4 insert_all/3 merge/3 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 size/2 subsumes/2 subterm/2 top/3 top_next/5 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

binary_heap_min

object

1.113.9 `pairing_heap(Order)`

- `Order` - Either `<` for a min heap or `>` for a max heap.

Pairing heap implementation, parameterized by the order to be used to compare keys (`<` or `>`).

Availability:
`logtalk_load(heaps(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-04-17

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *heap_protocol*
Extends:
`public` *compound*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_heap/2 as_list/2 check/1 delete/4 depth/2 empty/1 ground/1 insert/4 insert_all/3 merge/3 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 size/2 subsumes/2 subterm/2 top/3 top_next/5 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairing_heap_min, *pairing_heap_max*

object

1.113.10 `pairing_heap_max`

Max-pairing heap implementation. Uses standard order to compare keys.

Availability:
`logtalk_load(heaps(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-01-28

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *pairing_heap(>)*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_heap/2 as_list/2 check/1 delete/4 depth/2 empty/1 ground/1 insert/4 insert_all/3 merge/3 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 size/2 subsumes/2 subterm/2 top/3 top_next/5 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairing_heap_min, *pairing_heap(Order)*

object

1.113.11 `pairing_heap_min`

Min-pairing heap implementation. Uses standard order to compare keys.

Availability:
`logtalk_load(heaps(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-01-28

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *pairing_heap(<)*

Remarks:
(none)

Inherited public predicates:

(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_heap/2 as_list/2 check/1 delete/4 depth/2 empty/1 ground/1 insert/4 insert_all/3 merge/3 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 size/2 subsumes/2 subterm/2 top/3 top_next/5 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairing_heap_max, *pairing_heap(Order)*

1.114 help

object

1.114.1 `help`

Command-line help for Logtalk tools, libraries, entities, predicates, and non-terminals.

Availability:

```
logtalk_load(help(loader))
```

Author: Paulo Moura
Version: 0:44:0
Date: 2026-10-30

Compilation flags:

```
static, context_switching_calls, complements(allow)
```

Implements:

`public` *forwarding*

Uses:

atom
integer
list
os
user

Remarks:

(none)

Inherited public predicates:

forward/1

- *Public predicates*
  - `help/0`
  - `handbook/0`
  - `apis/0`
  - `apis/1`
  - `tools/0`
  - `tool/1`
  - `libraries/0`
  - `library/1`
  - `entity/1`
  - `(/)/2`
  - `(//)/2`
  - `man/1`
  - `completion/2`
  - `completions/2`
  - `built_in_directive/4`
  - `built_in_predicate/4`
  - `built_in_method/4`
  - `control_construct/4`
  - `built_in_non_terminal/4`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

help/0

Provides instructions on how to use the help tool.

Compilation flags:
`static`

Mode and number of proofs:
`help` - `one`

---

handbook/0

Provides access to the Handbook.

Compilation flags:
`static`

Mode and number of proofs:
`handbook` - `one`

---

apis/0

Provides access to the APIs documentation.

Compilation flags:
`static`

Mode and number of proofs:
`apis` - `one`

---

apis/1

Provides help on the given predicate or non-terminal.

Compilation flags:
`static`

Template:
`apis(Indicator)`
Mode and number of proofs:
`apis(+predicate_indicator)` - `one`
`apis(+non_terminal_indicator)` - `one`

---

tools/0

Provides access to the developer tools documentation.

Compilation flags:
`static`

Mode and number of proofs:
`tools` - `one`

---

tool/1

Provides help on the given developer tool.

Compilation flags:
`static`

Template:
`tool(Tool)`
Mode and number of proofs:
`tool(+atom)` - `zero_or_one`

---

libraries/0

Provides access to the standard libraries documentation.

Compilation flags:
`static`

Mode and number of proofs:
`libraries` - `one`

---

library/1

Provides help on the given standard library.

Compilation flags:
`static`

Template:
`library(Library)`
Mode and number of proofs:
`library(+atom)` - `zero_or_one`

---

entity/1

Provides help on the given built-in, tool, and library entity (object, protocol, or category).

Compilation flags:
`static`

Template:
`entity(Entity)`
Mode and number of proofs:
`entity(+entity_identifier)` - `zero_or_one`

(/)/2

Provides help on the `Name/Arity` built-in control construct, directive, predicate, or method.

Compilation flags:
`static`

Template:
`Name/Arity`
Mode and number of proofs:
`+atom/ +integer` - `zero_or_one`

(//)/2

Provides help on the `Name//Arity` built-in non-terminal.

Compilation flags:
`static`

Template:
`Name//Arity`
Mode and number of proofs:
`+atom// +integer` - `zero_or_one`

---

man/1

Opens the man page of the given script. On POSIX systems, the page is open inline. On Windows system, the HTML version of the man page is open on the operating-system default browser.

Compilation flags:
`static`

Template:
`man(Page)`
Mode and number of proofs:
`man(+atom)` - `one`

---

completion/2

Provides a completion pair, `Completion-Page`, for a given prefix.

Compilation flags:
`static`

Template:
`completion(Prefix,Completion)`
Mode and number of proofs:
`completion(+atom,-pair)` - `zero_or_more`

---

completions/2

Provides a list of completions pairs, `Completion-Page`, for a given prefix.

Compilation flags:
`static`

Template:
`completions(Prefix,Completions)`
Mode and number of proofs:
`completions(+atom,-lists(pair))` - `zero_or_more`

---

built_in_directive/4

Provides access to the HTML documenting files describing built-in directives.

Compilation flags:

```
static
```

Template:

```
built_in_directive(Name,Arity,Directory,Basename)
```

Mode and number of proofs:

```
built_in_directive(?atom,?integer,-atom,-atom) - zero_or_more
```

---

built_in_predicate/4

Provides access to the HTML documenting files describing built-in predicates.

Compilation flags:

```
static
```

Template:

```
built_in_predicate(Name,Arity,Directory,Basename)
```

Mode and number of proofs:

```
built_in_predicate(?atom,?integer,-atom,-atom) - zero_or_more
```

---

built_in_method/4

Provides access to the HTML documenting files describing built-in methods.

Compilation flags:

```
static
```

Template:

```
built_in_method(Name,Arity,Directory,Basename)
```

Mode and number of proofs:

```
built_in_method(?atom,?integer,-atom,-atom) - zero_or_more
```

---

control_construct/4

Provides access to the HTML documenting files describing built-in control constructs.

Compilation flags:
`static`

Template:
`control_construct(Name,Arity,Directory,Basename)`
Mode and number of proofs:
`control_construct(?atom,?integer,-atom,-atom)` - `zero_or_more`

---

built_in_non_terminal/4

Provides access to the HTML documenting files describing built-in DCG non-terminals.

Compilation flags:
`static`

Template:
`built_in_non_terminal(Name,Arity,Directory,Basename)`
Mode and number of proofs:
`built_in_non_terminal(?atom,?integer,-atom,-atom)` - `zero_or_more`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.115 hierarchical_clustering

object

1.115.1 hierarchical_clustering

Hierarchical clusterer for continuous datasets. Learns a full agglomerative_clusterer merge tree from a dataset object implementing the `clustering_dataset_protocol` protocol and then cuts the hierarchy to the requested number of clusters for prediction and export.

Availability:
`logtalk_load(hierarchical_clustering(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *clusterer_common*

Uses:
avltree
binary_heap_min
format
list
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *cut/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`cut/3`

Re-cuts a learned hierarchy to a new number of clusters without retraining.

Compilation flags:
```
static
```

Template:
```
cut(Clusterer,K,RecutClusterer)
```
Mode and number of proofs:
```
cut(+compound,+integer,-compound) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *agglomerative_clusterer*

1.116 hierarchies

category

1.116.1 `class_hierarchy`

Class hierarchy predicates.

Availability:
```
logtalk_load(hierarchies(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2006-02-20

Compilation flags:
```
static
```

Implements:

`public` *class_hierarchyp*

Remarks:
(none)

Inherited public predicates:
ancestor/1 *ancestors/1* *class/1* *classes/1* *descendant/1* *descendant_class/1* *descendant_classes/1* *descendant_instance/1* *descendant_instances/1* *descendants/1* *instance/1* *instances/1* *leaf/1* *leaf_class/1* *leaf_classes/1* *leaf_instance/1* *leaf_instances/1* *leaves/1* *subclass/1* *subclasses/1* *superclass/1* *superclasses/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.116.2 `class_hierarchyp`

Class hierarchy protocol.

Availability:
`logtalk_load(hierarchies(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2000-07-24

Compilation flags:
`static`

Extends:
`public` *hierarchyp*

Remarks:
(none)

Inherited public predicates:
ancestor/1 *ancestors/1* *descendant/1* *descendants/1* *leaf/1* *leaves/1*

- *Public predicates*
  - *class/1*
  - *classes/1*
  - *instance/1*
  - *instances/1*
  - *subclass/1*
  - *subclasses/1*
  - *superclass/1*
  - *superclasses/1*
  - *leaf_instance/1*
  - *leaf_instances/1*
  - *leaf_class/1*
  - *leaf_classes/1*
  - *descendant_instance/1*
  - *descendant_instances/1*
  - *descendant_class/1*
  - *descendant_classes/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

class/1

Returns, by backtracking, all object classes.

Compilation flags:
`static`

Template:
`class(Class)`

Mode and number of proofs:
`class(?object)` - `zero_or_more`

classes/1

List of all object classes.

Compilation flags:
`static`

Template:
`classes(Classes)`
Mode and number of proofs:
`classes(-list)` - `one`

---

instance/1

Returns, by backtracking, all class instances.

Compilation flags:
`static`

Template:
`instance(Instance)`
Mode and number of proofs:
`instance(?object)` - `zero_or_more`

---

instances/1

List of all class instances.

Compilation flags:
`static`

Template:
`instances(Instances)`
Mode and number of proofs:
`instances(-list)` - `one`

---

subclass/1

Returns, by backtracking, all class subclasses.

Compilation flags:
`static`

Template:
`subclass(Subclass)`
Mode and number of proofs:
`subclass(?object)` - `zero_or_more`

---

subclasses/1

List of all class subclasses.

Compilation flags:
`static`

Template:
`subclasses(Subclasses)`
Mode and number of proofs:
`subclasses(-list)` - `one`

---

superclass/1

Returns, by backtracking, all class superclasses.

Compilation flags:
`static`

Template:
`superclass(Superclass)`
Mode and number of proofs:
`superclass(?object)` - `zero_or_more`

---

superclasses/1

List of all class superclasses.

Compilation flags:
`static`

Template:
`superclasses(Superclasses)`
Mode and number of proofs:
`superclasses(-list)` - `one`

---

leaf_instance/1

Returns, by backtracking, all class leaf instances.

Compilation flags:
`static`

Template:
`leaf_instance(Leaf)`
Mode and number of proofs:
`leaf_instance(?object)` - `zero_or_more`

---

leaf_instances/1

List of all class leaf instances.

Compilation flags:
`static`

Template:
`leaf_instances(Leaves)`
Mode and number of proofs:
`leaf_instances(-list)` - `one`

---

leaf_class/1

Returns, by backtracking, all class leaf subclasses.

Compilation flags:
`static`

Template:
`leaf_class(Leaf)`
Mode and number of proofs:
`leaf_class(?object)` - `zero_or_more`

---

leaf_classes/1

List of all class leaf leaf subclasses.

Compilation flags:
`static`

Template:
`leaf_classes(Leaves)`
Mode and number of proofs:
`leaf_classes(-list)` - `one`

---

descendant_instance/1

Returns, by backtracking, all class descendant instances.

Compilation flags:
`static`

Template:
`descendant_instance(Descendant)`
Mode and number of proofs:
`descendant_instance(?object)` - `zero_or_more`

---

descendant_instances/1

List of all class descendant instances.

Compilation flags:
```
static
```

Template:
```
descendant_instances(Descendants)
```
Mode and number of proofs:
```
descendant_instances(-list) - one
```

---

descendant_class/1

Returns, by backtracking, all class descendant subclasses.

Compilation flags:
```
static
```

Template:
```
descendant_class(Descendant)
```
Mode and number of proofs:
```
descendant_class(?object) - zero_or_more
```

---

descendant_classes/1

List of all class descendant subclasses.

Compilation flags:
```
static
```

Template:
```
descendant_classes(Descendants)
```
Mode and number of proofs:
```
descendant_classes(-list) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

class_hierarchy

protocol

1.116.3 `hierarchyp`

Common hierarchy protocol for prototype and class hierarchies.

Availability:
`logtalk_load(hierarchies(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2000-07-24

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *ancestor/1*
  - *ancestors/1*
  - *leaf/1*
  - *leaves/1*
  - *descendant/1*
  - *descendants/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

ancestor/1

Returns, by backtracking, all object ancestors.

Compilation flags:
`static`

Template:
`ancestor(Ancestor)`
Mode and number of proofs:
`ancestor(?object)` - `zero_or_more`

---

ancestors/1

List of all object ancestors.

Compilation flags:
`static`

Template:
`ancestors(Ancestors)`
Mode and number of proofs:
`ancestors(-list)` - `one`

---

leaf/1

Returns, by backtracking, all object leaves.

Compilation flags:
`static`

Template:
`leaf(Leaf)`
Mode and number of proofs:
`leaf(?object)` - `zero_or_more`

---

leaves/1

List of all object leaves.

Compilation flags:
```
static
```

Template:
```
leaves(Leaves)
```
Mode and number of proofs:
```
leaves(-list) - one
```

---

descendant/1

Returns, by backtracking, all object descendants.

Compilation flags:
```
static
```

Template:
```
descendant(Descendant)
```
Mode and number of proofs:
```
descendant(?object) - zero_or_more
```

---

descendants/1

List of all object descendants.

Compilation flags:
```
static
```

Template:
```
descendants(Descendants)
```
Mode and number of proofs:
```
descendants(-list) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.116.4 `proto_hierarchy`

Prototype hierarchy predicates.

Availability:
`logtalk_load(hierarchies(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2006-02-20

Compilation flags:
`static`

Implements:
`public` *proto_hierarchyp*

Remarks:
(none)

Inherited public predicates:
ancestor/1 *ancestors/1* *descendant/1* *descendants/1* *extension/1* *extensions/1* *leaf/1* *leaves/1* *parent/1* *parents/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.116.5 `proto_hierarchyp`

Prototype hierarchy protocol.

Availability:
`logtalk_load(hierarchies(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2006-02-20

Compilation flags:
`static`

Extends:
`public` *hierarchyp*

Remarks:
(none)

Inherited public predicates:
ancestor/1 *ancestors/1* *descendant/1* *descendants/1* *leaf/1* *leaves/1*

- *Public predicates*
  - *parent/1*
  - *parents/1*
  - *extension/1*
  - *extensions/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parent/1

Returns, by backtracking, all object parents.

Compilation flags:
```
static
```

Template:
```
parent(Parent)
```
Mode and number of proofs:
```
parent(?object) - zero_or_more
```

---

parents/1

List of all object parents.

Compilation flags:
```
static
```

Template:
```
parents(Parents)
```
Mode and number of proofs:
```
parents(-list) - one
```

---

extension/1

Returns, by backtracking, all object direct descendants.

Compilation flags:
```
static
```

Template:
```
extension(Extension)
```
Mode and number of proofs:
```
extension(?object) - zero_or_more
```

`extensions/1`

List of all object direct descendants.

Compilation flags:
`static`

Template:
`extensions(Extensions)`
Mode and number of proofs:
`extensions(-list)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

proto_hierarchy

1.117 hmac

object

1.117.1 `hmac`

HMAC (Keyed-Hash Message Authentication Code) implementation as specified in RFC 2104.

Availability:
`logtalk_load(hmac(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-17

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *hmac_protocol*

Uses:

hash_common_32
list

Remarks:

(none)

Inherited public predicates:

digest/4 *digest/5* *hex_digest/4* *hex_digest/5*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

md5, *sha1*, *sha224*, *sha256*, *sha384*, *sha512*, *sha512_256*, *sha3_224*, *sha3_256*, *sha3_384*, *sha3_512*

protocol

1.117.2 `hmac_protocol`

Protocol for one-shot HMAC digest computation using hash objects implementing `hash_digest_protocol`.

Availability:

`logtalk_load(hmac(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-16

Compilation flags:

`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *digest/4*
  - *hex_digest/4*
  - *digest/5*
  - *hex_digest/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

digest/4

Computes the full HMAC digest for a hash object, key bytes, and message bytes.

Compilation flags:
`static`

Template:
`digest(Hash,KeyBytes,MessageBytes,DigestBytes)`
Mode and number of proofs:
`digest(+object_identifier,+list(byte),+list(byte),--list(byte))` - `one`

---

hex_digest/4

Computes the full HMAC digest for a hash object, key bytes, and message bytes, returning a lowercase hexadecimal atom.

Compilation flags:
`static`

Template:
`hex_digest(Hash,KeyBytes,MessageBytes,HexDigest)`
Mode and number of proofs:
`hex_digest(+object_identifier,+list(byte),+list(byte),--atom)` - `one`

---

digest/5

Computes a truncated HMAC digest containing the requested number of leftmost bytes.

Compilation flags:

```
static
```

Template:

```
digest(Hash,KeyBytes,MessageBytes,Length,DigestBytes)
```

Mode and number of proofs:

```
digest(+object_identifier,+list(byte),+list(byte),+integer,--list(byte)) - one
```

---

hex_digest/5

Computes a truncated HMAC digest containing the requested number of leftmost bytes and returns it as a lowercase hexadecimal atom.

Compilation flags:

```
static
```

Template:

```
hex_digest(Hash,KeyBytes,MessageBytes,Length,HexDigest)
```

Mode and number of proofs:

```
hex_digest(+object_identifier,+list(byte),+list(byte),+integer,--atom) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

hash_digest_protocol

1.118 hodge_rank

object

1.118.1 `hodge_rank`

HodgeRank pairwise measurement ranker. Learns one deterministic zero-sum score per item from a dataset object implementing the `pairwise_measurement_dataset_protocol` protocol by solving the weighted graph-Laplacian least-squares system induced by signed edge measurements and returns a self-describing ranker term with diagnostics and residuals.

Availability:

`logtalk_load(hodge_rank(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-08-18

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *ranking_dataset_common*
`public` *score_ranker_model_common*
`public` *pairwise_strength_ranker_common*

Uses:

avltree
list
numberlist

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
  - *residuals/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

residuals/2

Returns the learned weighted edge residuals as `residual(Item1,Item2,Residual,Weight)` terms preserving measurement enumeration order.

Compilation flags:
`static`

Template:
`residuals(Ranker,Residuals)`

Mode and number of proofs:
`residuals(+compound,-list(compound))` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_measurement_dataset_protocol, *ranker_protocol*, *massey_ranker*

1.119 hook_flows

object

1.119.1 `hook_pipeline(Pipeline)`

- `Pipeline` - List of hook objects.

Use a pipeline (represented using a list) of hook objects to expand terms and goals. The expansion results from a hook object are passed to the next hook object in the pipeline.

Availability:
`logtalk_load(hook_flows(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2024-09-27

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Remarks:

- Usage: Compile source files that should be expanded using the pipeline of hook objects using the compiler option `hook(hook_pipeline(Pipeline))`.

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

hook_set(Set)

object

1.119.2 `hook_set(Set)`

- `Set` - Set (list) of hook objects.

Use a set (represented using a list) of hook objects to expand terms and goals. The hook objects are tried in sequence until one of them succeeds in expanding the current term (goal) into a different term (goal).

Availability:

```
logtalk_load(hook_flows(loader))
```

Author: Paulo Moura
Version: 2:0:0
Date: 2024-09-27

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *expanding*

Remarks:

- Usage: Compile source files that should be expanded using the set of hook objects using the compiler option `hook(hook_set(Set))`.

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

hook_pipeline(Pipeline)

1.120 hook_objects

object

1.120.1 `backend_adapter_hook`

This hook object applies the expansion rules defined in the Prolog backend adapter file.

Availability:
`logtalk_load(hook_objects(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2020-02-17

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

default_workflow_hook, *identity_hook*, *grammar_rules_hook*, *prolog_module_hook(Module)*, *object_wrapper_hook*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.2 `default_workflow_hook`

Use this object as the default hook object to restore the default expansion pipeline semantics used by the compiler.

Availability:
`logtalk_load(hook_objects(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2020-03-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

backend_adapter_hook, *identity_hook*, *grammar_rules_hook*, *prolog_module_hook(Module)*, *object_wrapper_hook*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.3 `grammar_rules_hook`

This hook object expands grammar rules into clauses.

Availability:
`logtalk_load(hook_objects(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2020-02-14

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Remarks:
(none)

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

backend_adapter_hook, *default_workflow_hook*, *identity_hook*, *prolog_module_hook(Module)*, *object_wrapper_hook*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.4 `identity_hook`

Use this object as a file specific hook object to prevent any (other) user-defined expansion rules to be applied when compiling the file.

Availability:

```
logtalk_load(hook_objects(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2020-02-15

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public expanding
```

Remarks:

(none)

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

backend_adapter_hook, *default_workflow_hook*, *grammar_rules_hook*, *prolog_module_hook(Module)*, *object_wrapper_hook*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.5 `object_wrapper_hook`

Use this object to wrap the contents of a plain Prolog file in an object named after the file. The wrapper sets the `context_switching_calls` flag to `allow`, enabling calling of the wrapped predicates using the `<</2` control construct.

Availability:

`logtalk_load(hook_objects(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2020-10-30

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *expanding*

Uses:

os

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

object_wrapper_hook(Protocol), *object_wrapper_hook(Name,Relations)*, *backend_adapter_hook*, *default_workflow_hook*, *grammar_rules_hook*, *prolog_module_hook(Module)*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.6 `object_wrapper_hook(Protocol)`

Use this object to wrap the contents of a plain Prolog file in an object named after the file that implements the given protocol.

Availability:
```
logtalk_load(hook_objects(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2021-11-24

Compilation flags:
```
static, context_switching_calls
```

Implements:

`public` *expanding*

Uses:

os

Remarks:

(none)

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

object_wrapper_hook, *object_wrapper_hook(Name,Relations)*, *backend_adapter_hook*, *default_workflow_hook*, *grammar_rules_hook*, *prolog_module_hook(Module)*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.7 `object_wrapper_hook(Name,Relations)`

Use this object to wrap the contents of a plain Prolog file in an object with the given name and object entity relations (a list).

Availability:

`logtalk_load(hook_objects(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2022-02-03

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *expanding*

Remarks:

(none)

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

object_wrapper_hook, *object_wrapper_hook(Protocol)*, *backend_adapter_hook*, *default_workflow_hook*, *grammar_rules_hook*, *prolog_module_hook(Module)*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.8 print_goal_hook

Use this object to easily print entity predicate goals before, after, or before and after calling them.

Availability:
`logtalk_load(hook_objects(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2020-03-14

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Remarks:

- Usage: Mark a goal to be printed by prefixing it with an operator. Printing uses a `comment` message.
- To print goal before calling it: `- Goal`.
- To print goal after calling it: `+ Goal`.
- To print goal before and after calling it: `* Goal`.
- Operators: This hook object uses the standard `-` and `+` prefix operators and also defines a global `*` prefix operator with the same type and priority.

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

backend_adapter_hook, *default_workflow_hook*, *grammar_rules_hook*, *identity_hook*, *prolog_module_hook(Module)*, *object_wrapper_hook*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *suppress_goal_hook*

object

1.120.9 `prolog_module_hook(Module)`

This hook object applies the expansion rules defined in a Prolog module (e.g., `user`).

Availability:
`logtalk_load(hook_objects(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2020-02-17

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

backend_adapter_hook, *default_workflow_hook*, *identity_hook*, *grammar_rules_hook*, *object_wrapper_hook*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.10 `suppress_goal_hook`

Use this object to easily suppress a goal in a clause body.

Availability:
`logtalk_load(hook_objects(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2020-05-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Remarks:

- Usage: Mark a goal to be suppressed by prefixing it with the `--` operator.
- Operators: This hook object uses the `--` prefix operator declared by Logtalk for use in `mode/2` directives.

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

backend_adapter_hook, *default_workflow_hook*, *grammar_rules_hook*, *identity_hook*, *prolog_module_hook(Module)*, *object_wrapper_hook*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*

object

1.120.11 `write_to_file_hook(File)`

This hook object writes term-expansion results to a file in canonical format. The terms are terminated by a period and a new line.

Availability:
`logtalk_load(hook_objects(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2022-07-06

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *write_to_file_hook(File,[quoted(true),ignore_ops(true)])*

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

backend_adapter_hook, *default_workflow_hook*, *identity_hook*, *grammar_rules_hook*, *prolog_module_hook(Module)*, *object_wrapper_hook*, *write_to_file_hook(File,Options)*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.12 `write_to_file_hook(File,Options)`

This hook object writes term-expansion results to a file using a list of `write_term/3` options. The terms are terminated by a period and a new line.

Availability:
`logtalk_load(hook_objects(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2022-07-06

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Remarks:
(none)

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

backend_adapter_hook, *default_workflow_hook*, *identity_hook*, *grammar_rules_hook*, *prolog_module_hook(Module)*, *object_wrapper_hook*, *write_to_file_hook(File)*, *write_to_stream_hook(Stream,Options)*, *write_to_stream_hook(Stream)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.13 `write_to_stream_hook(Stream)`

This hook object writes term-expansion results to a stream in canonical format. The terms are terminated by a period and a new line.

Availability:

```
logtalk_load(hook_objects(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2020-02-16

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public write_to_stream_hook(Stream,[quoted(true),ignore_ops(true)])
```

Remarks:

(none)

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

backend_adapter_hook, *default_workflow_hook*, *identity_hook*, *grammar_rules_hook*, *prolog_module_hook(Module)*, *object_wrapper_hook*, *write_to_stream_hook(Stream,Options)*, *write_to_file_hook(File,Options)*, *write_to_file_hook(File)*, *print_goal_hook*, *suppress_goal_hook*

object

1.120.14 `write_to_stream_hook(Stream,Options)`

This hook object writes term-expansion results to a stream using a list of `write_term/3` options. The terms are terminated by a period and a new line.

Availability:

```
logtalk_load(hook_objects(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2020-02-16

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *expanding*

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

backend_adapter_hook, *default_workflow_hook*, *identity_hook*, *grammar_rules_hook*, *prolog_module_hook(Module)*, *object_wrapper_hook*, *write_to_stream_hook(Stream)*, *write_to_file_hook(File,Options)*, *write_to_file_hook(File)*, *print_goal_hook*, *suppress_goal_hook*

1.121 html

category

1.121.1 `html`

HTML generation.

Availability:
`logtalk_load(html(loader))`

Author: Paul Brown and Paulo Moura
Version: 0:5:1
Date: 2026-11-11

Compilation flags:
`static`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *generate/2*
  - *void_element/1*
  - *normal_element/2*
  - *aggregate_resources/2*
- *Protected predicates*
- *Private predicates*
  - *doctype/1*
- *Operators*

Public predicates

generate/2

Generates HTML content using the representation specified in the first argument (`stream(Stream)` or `file(Path)`) for the term in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,Term)`

Mode and number of proofs:
`generate(+compound,++term)` - `one_or_error`

Exceptions:
`Sink` is not a ground term:
`instantiation_error`
`Term` is not a ground term:
`instantiation_error`
`Sink` is ground but not a valid sink term:
`domain_error(html_sink,Sink)`

void_element/1

Enumerates, by backtracking, all void elements.

Compilation flags:
`static`

Template:
`void_element(Element)`
Mode and number of proofs:
`void_element(?atom)` - `zero_or_more`

---

normal_element/2

Enumerates, by backtracking, all normal elements. The value of the `Display` argument is either `inline` or `block`.

Compilation flags:
`static`

Template:
`normal_element(Element,Display)`
Mode and number of proofs:
`normal_element(?atom,?atom)` - `zero_or_more`

---

aggregate_resources/2

Returns a dependency-aware, deduplicated list of resource declarations.

Compilation flags:
`static`

Template:
`aggregate_resources(Declarations,Resources)`
Mode and number of proofs:
`aggregate_resources(+list,-list)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

doctype/1

Doctype text.

Compilation flags:
`static`

Template:
`doctype(DocType)`
Mode and number of proofs:
`doctype(?atom)` - `one`

Operators

(none)
object

1.121.2 html5

HTML content generation using the HTML 5 doctype.

Availability:
`logtalk_load(html(loader))`

Author: Paul Brown and Paulo Moura
Version: 1:0:0
Date: 2021-03-29

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *html*

Remarks:
(none)

Inherited public predicates:
aggregate_resources/2 *generate/2* *normal_element/2* *void_element/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.121.3 `xhtml11`

XHTML content generation using the XHTML 1.1 doctype.

Availability:
`logtalk_load(html(loader))`

Author: Paul Brown and Paulo Moura
Version: 1:0:0
Date: 2021-03-29

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *html*

Remarks:
(none)

Inherited public predicates:
aggregate_resources/2 *generate/2* *normal_element/2* *void_element/1*

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.122 http_authenticate

object

1.122.1 `http_authenticate`

HTTP Basic authentication parsing, generation, challenge building, and request verification helpers.

Availability:
`logtalk_load(http_authenticate(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-09-08

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
`public` *http_authentication_helpers*

Uses:
base64
http_core
list
type
user

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *challenge/2*
  - *authorization/2*
  - *parse_challenge/2*
  - *generate_challenge/2*
  - *parse_authorization/2*
  - *generate_authorization/2*
  - *protect_request/4*
  - *unauthorized_response/3*
  - *unauthorized_response/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

challenge/2

Returns the single parsed Basic `WWW-Authenticate` challenge from a normalized HTTP response when present.

Compilation flags:

`static`

Template:

`challenge(Response,Challenge)`

Mode and number of proofs:

`challenge(+compound,-compound)` - `zero_or_one`

---

authorization/2

Returns the single parsed Basic `Authorization` header from a normalized HTTP request when present.

Compilation flags:

`static`

Template:

`authorization(Request,Authorization)`

Mode and number of proofs:

`authorization(+compound,-compound)` - `zero_or_one`

---

parse_challenge/2

Parses one Basic challenge header field value into a normalized `basic_challenge/1` term.

Compilation flags:
`static`

Template:
`parse_challenge(Text,Challenge)`
Mode and number of proofs:
`parse_challenge(++text,-compound)` - `one_or_error`

Exceptions:
`Text` is not a valid Basic challenge header value:
`domain_error(http_authenticate_header(www_authenticate),invalid(syntax))`
`Text` uses an unsupported authentication scheme:
`domain_error(http_authenticate_header(www_authenticate),`
`unsupported_scheme(Scheme))`
`Text` contains a duplicated Basic challenge directive:
`domain_error(http_authenticate_header(www_authenticate),duplicate(Name))`
`Text` is missing a required Basic challenge directive:
`domain_error(http_authenticate_header(www_authenticate),missing(Name))`
`Text` contains an unexpected Basic challenge directive:
`domain_error(http_authenticate_header(www_authenticate),unexpected(Name))`
`Text` contains an invalid Basic challenge charset:
`domain_error(http_authenticate_header(www_authenticate),invalid(charset))`

---

generate_challenge/2

Generates one canonical Basic challenge header field value from a normalized `basic_challenge/1` term.

Compilation flags:
`static`

Template:
`generate_challenge(Challenge,HeaderValue)`
Mode and number of proofs:
`generate_challenge(+compound,-atom)` - `one_or_error`

Exceptions:
`Challenge` is not a valid normalized Basic challenge term:
`domain_error(http_authenticate_term(challenge),Challenge)`
`Challenge` is missing a required field:
`domain_error(http_authenticate_term(challenge),missing(Name))`
`Challenge` contains a duplicated field:
`domain_error(http_authenticate_term(challenge),duplicate(Name))`
`Challenge` contains an unexpected field:

`domain_error(http_authenticate_term(challenge),unexpected(Name))`
`Challenge` contains an invalid field:
`domain_error(http_authenticate_term(challenge),invalid(Field))`

---

parse_authorization/2

Parses one Basic authorization header field value into a normalized `basic_authorization/1` term.

Compilation flags:
`static`

Template:
`parse_authorization(Text,Authorization)`
Mode and number of proofs:
`parse_authorization(++text,-compound)` - `one_or_error`

Exceptions:
`Text` is not a valid Basic authorization header value:
`domain_error(http_authenticate_header(authorization),invalid(syntax))`
`Text` uses an unsupported authentication scheme:
`domain_error(http_authenticate_header(authorization),`
`unsupported_scheme(Scheme))`
`Text` contains invalid Basic credentials:
`domain_error(http_authenticate_header(authorization),invalid(credentials))`

---

generate_authorization/2

Generates one canonical Basic authorization header field value from a normalized `basic_authorization/1` term.

Compilation flags:
`static`

Template:
`generate_authorization(Authorization,HeaderValue)`
Mode and number of proofs:
`generate_authorization(+compound,-atom)` - `one_or_error`

Exceptions:
`Authorization` is not a valid normalized Basic authorization term:
`domain_error(http_authenticate_term(authorization),Authorization)`
`Authorization` contains an invalid Basic username:
`domain_error(http_authenticate_value(username),Username)`

---

protect_request/4

Verifies a normalized HTTP request using a Basic verifier object and returns either `continue(Request)` or `respond(Response)`.

Compilation flags:

`static`

Template:

`protect_request(Request,Verifier,Action,Options)`

Mode and number of proofs:

`protect_request(+compound,+object_identifier,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`Verifier` is a variable:

`instantiation_error`

`Verifier` is neither a variable nor an existing object:

`existence_error(http_authenticate_verifier,Verifier)`

`Verifier` does not implement the Basic verifier protocol:

`domain_error(http_authenticate_verifier,Verifier)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid Basic-auth status:

`domain_error(http_authenticate_status,Status)`

`Options` contains invalid Basic-auth headers:

`domain_error(http_headers,Headers)`

`Options` contains an invalid Basic-auth body:

`domain_error(http_body,Body)`

`Options` contains invalid Basic-auth properties:

`domain_error(http_properties,Properties)`

unauthorized_response/3

Builds a normalized `401 Unauthorized` response and returns the generated normalized Basic challenge term.

Compilation flags:

`static`

Template:

`unauthorized_response(Challenge,Response,Options)`

Mode and number of proofs:

`unauthorized_response(-compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid Basic-auth status:

`domain_error(http_authenticate_status,Status)`

`Options` contains invalid Basic-auth headers:

`domain_error(http_headers,Headers)`

`Options` contains an invalid Basic-auth body:

`domain_error(http_body,Body)`

`Options` contains invalid Basic-auth properties:

`domain_error(http_properties,Properties)`

The generated response headers violate normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

The generated response properties violate normalized HTTP response semantics:

`domain_error(http_property_semantics,Property)`

---

unauthorized_response/4

Decorates a normalized HTTP response with an explicit normalized Basic challenge term and returns the resulting `401 Unauthorized` response.

Compilation flags:

`static`

Template:

`unauthorized_response(Challenge,Response0,Response,Options)`

Mode and number of proofs:

`unauthorized_response(+compound,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Challenge` is not a valid normalized Basic challenge term:
`domain_error(http_authenticate_term(challenge),Challenge)`
`Response0` is not a valid normalized HTTP response term:
`domain_error(http_response,Response0)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid unauthorized-response overlay option:
`domain_error(http_authenticate_unauthorized_response_overlay_option,Option)`
`Options` contains an invalid Basic-auth status:
`domain_error(http_authenticate_status,Status)`
`Options` contains invalid Basic-auth headers:
`domain_error(http_headers,Headers)`
`Options` contains an invalid Basic-auth body:
`domain_error(http_body,Body)`
`Options` contains invalid Basic-auth properties:
`domain_error(http_properties,Properties)`
The decorated response headers violate normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`
The decorated response properties violate normalized HTTP response semantics:
`domain_error(http_property_semantics,Property)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.122.2 `http_authenticate_verifier_protocol`

Protocol for HTTP Basic authentication verifiers backed by application-defined credential stores.

Availability:

`logtalk_load(http_authenticate(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-05-29

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *verify/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

verify/3

Succeeds when the plaintext password is valid for the given realm and username.

Compilation flags:
`static`

Template:
`verify(Realm,Username,Password)`
Mode and number of proofs:
`verify(++atom,++atom,++atom) - zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.122.3 `http_htpasswd_verifier(Path)`

- `Path` - Path of the password file to load on each verification request.

Portable Apache `.htpasswd` subset verifier supporting `{SHA}`, `$apr1$`, and bcrypt `$2b$` entries and rejecting unsupported hash markers.

Availability:
`logtalk_load(http_authenticate(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-08-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_authenticate_verifier_protocol*
Imports:
`public` *http_text_helpers*
Uses:
base64
crypto
list
os
reader

Remarks:
(none)

Inherited public predicates:
verify/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.122.4 `http_router_basic_auth(Verifier,BaseOptions)`

- `Verifier` - Verifier object passed to `http_authenticate::protect_request/4` when a routed request declares Basic auth metadata.
- `BaseOptions` - Base protect-request options merged with any route-specific `basic_auth/1` metadata options.

Optional Basic-auth router helpers for objects importing the `http_router` category.

Availability:
`logtalk_load(http_authenticate(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static`

Uses:
http_authenticate
http_core
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *authorize_basic_auth_request/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

authorize_basic_auth_request/2

Router helper that applies Basic-auth route metadata to a routed request and returns either `continue(Request)` or `respond(Response)`.

Compilation flags:

`static`

Template:

`authorize_basic_auth_request(Request,Action)`

Mode and number of proofs:

`authorize_basic_auth_request(+compound,-compound)` - `one_or_error`

Exceptions:

Route metadata options are not a proper list:

`domain_error(http_router_basic_auth_options,Options)`

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

The Basic verifier object is invalid:

`domain_error(http_authenticate_verifier,Verifier)`

The Basic verifier object does not exist:

`existence_error(http_authenticate_verifier,Verifier)`

The Basic-auth protection options are invalid:

`domain_error(option,Option)`

The Basic-auth status is invalid:

`domain_error(http_authenticate_status,Status)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.122.5 `http_server_core_basic_handler(Verifier,Handler,ProtectOptions)`

- `Verifier` - Verifier object implementing the `http_authenticate_verifier_protocol` protocol.
- `Handler` - Wrapped object implementing the `http_handler_protocol` protocol.
- `ProtectOptions` - Options passed to `http_authenticate::protect_request/4`.

Wrapper HTTP handler object that applies Basic request verification around another portable HTTP handler.

Availability:
`logtalk_load(http_authenticate(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_handler_protocol*
Uses:
http_authenticate

Remarks:
(none)

Inherited public predicates:
handle/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.123 http_client

object

1.123.1 `http_client`

Request-oriented HTTP client facade built on top of the URL and HTTP transport libraries.

Availability:
`logtalk_load(http_client(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-10-20

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
`public` *http_message_helpers*
`public` *http_text_helpers*
`public` *http_origin_site_helpers*

Uses:
http_client_core
http_core
http_multipart
http_websocket_handshake
list
url(Representation)
user
uuid(Representation)

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*

    - request/4
    - request/5
    - get/3
    - get/4
    - head/3
    - head/4
    - delete/3
    - delete/4
    - post/4
    - post/5
    - put/4
    - put/5
    - patch/4
    - patch/5
    - query/4
    - query/5
    - open_websocket/4
    - open_sse/4
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

request/4

Builds a normalized request from the given method, absolute URL, and options, selects a compatible transport, performs a one-shot exchange, and returns the response.

Compilation flags:

`static`

Template:

`request(Method,URL,Response,Options)`

Mode and number of proofs:

`request(+atom,+atom,--compound,+list)` - `one_or_error`

Exceptions:

`URL` is a variable:

`instantiation_error`

`URL` is not a supported absolute HTTP URL:

`domain_error(http_client_url,URL)`

`URL` uses an unsupported HTTP scheme:

`domain_error(http_client_scheme,Scheme)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Options` contains an invalid HTTP client request option:
`domain_error(http_client_request_option,Option)`
`Options` contains invalid form-data headers:
`domain_error(http_client_form_data_headers,Headers)`
`Options` contains invalid form-data properties:
`domain_error(http_client_form_data_properties,Properties)`
The generated request is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
The delegated socket exchange rejects the response stream:
`domain_error(http_response_stream,Error)`

---

request/5

Builds a normalized request from the given method, absolute URL, and options, validates it against an open compatible connection or pool handle endpoint, selects a compatible transport, performs one exchange, and returns the response.

Compilation flags:
`static`

Template:
`request(ConnectionOrPool,Method,URL,Response,Options)`
Mode and number of proofs:
`request(+compound,+atom,+atom,--compound,+list)` - `one_or_error`

Exceptions:
`ConnectionOrPool` is a variable:
`instantiation_error`
`ConnectionOrPool` is not a valid reusable connection or pool handle:
`domain_error(http_socket_transport_connection_or_pool,ConnectionOrPool)`
`ConnectionOrPool` is not connected to the requested endpoint:
`domain_error(http_client_connection_target,endpoint(EndpointHost,EndpointPort,Host,Port))`
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`URL` uses an unsupported HTTP scheme:
`domain_error(http_client_scheme,Scheme)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid HTTP client request option:

`domain_error(http_client_request_option,Option)`

`Options` contains invalid form-data headers:

`domain_error(http_client_form_data_headers,Headers)`

`Options` contains invalid form-data properties:

`domain_error(http_client_form_data_properties,Properties)`

The generated request is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

The delegated socket exchange rejects the response stream:

`domain_error(http_response_stream,Error)`

`get/3`

Convenience wrapper over `request/4` using the `get` method.

Compilation flags:

`static`

Template:

`get(URL,Response,Options)`

Mode and number of proofs:

`get(+atom,--compound,+list)` - `one_or_error`

Exceptions:

`URL` is a variable:

`instantiation_error`

`URL` is not a supported absolute HTTP URL:

`domain_error(http_client_url,URL)`

`URL` uses an unsupported HTTP scheme:

`domain_error(http_client_scheme,Scheme)`

`Options` contains an invalid HTTP client request option:

`domain_error(http_client_request_option,Option)`

The delegated socket exchange rejects the response stream:

`domain_error(http_response_stream,Error)`

---

get/4

Convenience wrapper over `request/5` using the `get` method.

Compilation flags:
`static`

Template:
`get(ConnectionOrPool,URL,Response,Options)`
Mode and number of proofs:
`get(+compound,+atom,--compound,+list)` - `one_or_error`

Exceptions:
`ConnectionOrPool` is not a valid reusable connection or pool handle:
`domain_error(http_socket_transport_connection_or_pool,ConnectionOrPool)`
`ConnectionOrPool` is not connected to the requested endpoint:
`domain_error(http_client_connection_target,endpoint(EndpointHost,EndpointPort,Host,Port))`
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`URL` uses an unsupported HTTP scheme:
`domain_error(http_client_scheme,Scheme)`
`Options` contains an invalid HTTP client request option:
`domain_error(http_client_request_option,Option)`
The delegated socket exchange rejects the response stream:
`domain_error(http_response_stream,Error)`

---

head/3

Convenience wrapper over `request/4` using the `head` method.

Compilation flags:
`static`

Template:
`head(URL,Response,Options)`
Mode and number of proofs:
`head(+atom,--compound,+list)` - `one_or_error`

Exceptions:
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`URL` uses an unsupported HTTP scheme:

`domain_error(http_client_scheme,Scheme)`
`Options` contains an invalid HTTP client request option:
`domain_error(http_client_request_option,Option)`
The delegated socket exchange rejects the response stream:
`domain_error(http_response_stream,Error)`

---

head/4

Convenience wrapper over `request/5` using the `head` method.

Compilation flags:
`static`

Template:
`head(ConnectionOrPool,URL,Response,Options)`
Mode and number of proofs:
`head(+compound,+atom,--compound,+list)` - `one_or_error`

Exceptions:
`ConnectionOrPool` is not a valid reusable connection or pool handle:
`domain_error(http_socket_transport_connection_or_pool,ConnectionOrPool)`
`ConnectionOrPool` is not connected to the requested endpoint:
`domain_error(http_client_connection_target,endpoint(EndpointHost,EndpointPort,Host,Port))`
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`URL` uses an unsupported HTTP scheme:
`domain_error(http_client_scheme,Scheme)`
`Options` contains an invalid HTTP client request option:
`domain_error(http_client_request_option,Option)`
The delegated socket exchange rejects the response stream:
`domain_error(http_response_stream,Error)`

---

delete/3

Convenience wrapper over `request/4` using the `delete` method.

Compilation flags:
`static`

Template:
`delete(URL,Response,Options)`
Mode and number of proofs:

`delete(+atom,--compound,+list)` - `one_or_error`

Exceptions:

`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`URL` uses an unsupported HTTP scheme:
`domain_error(http_client_scheme,Scheme)`
`Options` contains an invalid HTTP client request option:
`domain_error(http_client_request_option,Option)`
The delegated socket exchange rejects the response stream:
`domain_error(http_response_stream,Error)`

`delete/4`

Convenience wrapper over `request/5` using the `delete` method.

Compilation flags:
`static`

Template:
`delete(ConnectionOrPool,URL,Response,Options)`
Mode and number of proofs:
`delete(+compound,+atom,--compound,+list)` - `one_or_error`

Exceptions:

`ConnectionOrPool` is not a valid reusable connection or pool handle:
`domain_error(http_socket_transport_connection_or_pool,ConnectionOrPool)`
`ConnectionOrPool` is not connected to the requested endpoint:
`domain_error(http_client_connection_target,endpoint(EndpointHost,EndpointPort,Host,Port))`
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`URL` uses an unsupported HTTP scheme:
`domain_error(http_client_scheme,Scheme)`
`Options` contains an invalid HTTP client request option:
`domain_error(http_client_request_option,Option)`
The delegated socket exchange rejects the response stream:
`domain_error(http_response_stream,Error)`

---

post/4

Convenience wrapper over `request/4` using the `post` method and the given request body.

Compilation flags:
`static`

Template:
`post(URL,Body,Response,Options)`
Mode and number of proofs:
`post(+atom,+compound,--compound,+list)` - `one_or_error`

Exceptions:
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`URL` uses an unsupported HTTP scheme:
`domain_error(http_client_scheme,Scheme)`
`Options` contains an invalid HTTP client request option:
`domain_error(http_client_request_option,Option)`
`Body` is invalid for the generated normalized HTTP request:
`domain_error(http_body,Body)`
The delegated socket exchange rejects the response stream:
`domain_error(http_response_stream,Error)`

---

post/5

Convenience wrapper over `request/5` using the `post` method and the given request body.

Compilation flags:
`static`

Template:
`post(ConnectionOrPool,URL,Body,Response,Options)`
Mode and number of proofs:
`post(+compound,+atom,+compound,--compound,+list)` - `one_or_error`

Exceptions:
`ConnectionOrPool` is not a valid reusable connection or pool handle:
`domain_error(http_socket_transport_connection_or_pool,ConnectionOrPool)`
`ConnectionOrPool` is not connected to the requested endpoint:
`domain_error(http_client_connection_target,endpoint(EndpointHost,EndpointPort,Host,Port))`
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute HTTP URL:

domain_error(http_client_url,URL)
URL uses an unsupported HTTP scheme:
domain_error(http_client_scheme,Scheme)
Options contains an invalid HTTP client request option:
domain_error(http_client_request_option,Option)
Body is invalid for the generated normalized HTTP request:
domain_error(http_body,Body)
The delegated socket exchange rejects the response stream:
domain_error(http_response_stream,Error)

---

put/4

Convenience wrapper over request/4 using the put method and the given request body.

Compilation flags:
static

Template:
put(URL,Body,Response,Options)
Mode and number of proofs:
put(+atom,+compound,--compound,+list) - one_or_error

Exceptions:
URL is a variable:
instantiation_error
URL is not a supported absolute HTTP URL:
domain_error(http_client_url,URL)
URL uses an unsupported HTTP scheme:
domain_error(http_client_scheme,Scheme)
Options contains an invalid HTTP client request option:
domain_error(http_client_request_option,Option)
Body is invalid for the generated normalized HTTP request:
domain_error(http_body,Body)
The delegated socket exchange rejects the response stream:
domain_error(http_response_stream,Error)

---

put/5

Convenience wrapper over request/5 using the put method and the given request body.

Compilation flags:
static

Template:
put(ConnectionOrPool,URL,Body,Response,Options)

Mode and number of proofs:

`put(+compound,+atom,+compound,--compound,+list)` - `one_or_error`

Exceptions:

`ConnectionOrPool` is not a valid reusable connection or pool handle:

`domain_error(http_socket_transport_connection_or_pool,ConnectionOrPool)`

`ConnectionOrPool` is not connected to the requested endpoint:

`domain_error(http_client_connection_target,endpoint(EndpointHost,EndpointPort,Host,Port))`

`URL` is a variable:

`instantiation_error`

`URL` is not a supported absolute HTTP URL:

`domain_error(http_client_url,URL)`

`URL` uses an unsupported HTTP scheme:

`domain_error(http_client_scheme,Scheme)`

`Options` contains an invalid HTTP client request option:

`domain_error(http_client_request_option,Option)`

`Body` is invalid for the generated normalized HTTP request:

`domain_error(http_body,Body)`

The delegated socket exchange rejects the response stream:

`domain_error(http_response_stream,Error)`

`patch/4`

Convenience wrapper over `request/4` using the `patch` method and the given request body.

Compilation flags:

`static`

Template:

`patch(URL,Body,Response,Options)`

Mode and number of proofs:

`patch(+atom,+compound,--compound,+list)` - `one_or_error`

Exceptions:

`URL` is a variable:

`instantiation_error`

`URL` is not a supported absolute HTTP URL:

`domain_error(http_client_url,URL)`

`URL` uses an unsupported HTTP scheme:

`domain_error(http_client_scheme,Scheme)`

`Options` contains an invalid HTTP client request option:

`domain_error(http_client_request_option,Option)`

`Body` is invalid for the generated normalized HTTP request:

`domain_error(http_body,Body)`

The delegated socket exchange rejects the response stream:

`domain_error(http_response_stream,Error)`

patch/5

Convenience wrapper over `request/5` using the `patch` method and the given request body.

Compilation flags:
`static`

Template:
`patch(ConnectionOrPool,URL,Body,Response,Options)`
Mode and number of proofs:
`patch(+compound,+atom,+compound,--compound,+list)` - `one_or_error`

Exceptions:
`ConnectionOrPool` is not a valid reusable connection or pool handle:
`domain_error(http_socket_transport_connection_or_pool,ConnectionOrPool)`
`ConnectionOrPool` is not connected to the requested endpoint:
`domain_error(http_client_connection_target,endpoint(EndpointHost,EndpointPort,Host,Port))`
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`URL` uses an unsupported HTTP scheme:
`domain_error(http_client_scheme,Scheme)`
`Options` contains an invalid HTTP client request option:
`domain_error(http_client_request_option,Option)`
`Body` is invalid for the generated normalized HTTP request:
`domain_error(http_body,Body)`
The delegated socket exchange rejects the response stream:
`domain_error(http_response_stream,Error)`

query/4

Convenience wrapper over `request/4` using the `query` method and the given request body.

Compilation flags:
`static`

Template:
`query(URL,Body,Response,Options)`
Mode and number of proofs:
`query(+atom,+compound,--compound,+list)` - `one_or_error`

Exceptions:
`URL` is a variable:

instantiation_error
URL is not a supported absolute HTTP URL:
domain_error(http_client_url,URL)
URL uses an unsupported HTTP scheme:
domain_error(http_client_scheme,Scheme)
Options contains an invalid HTTP client request option:
domain_error(http_client_request_option,Option)
Body is invalid for the generated normalized HTTP request:
domain_error(http_body,Body)
The delegated socket exchange rejects the response stream:
domain_error(http_response_stream,Error)

---

query/5

Convenience wrapper over request/5 using the query method and the given request body.

Compilation flags:
static

Template:
query(ConnectionOrPool,URL,Body,Response,Options)
Mode and number of proofs:
query(+compound,+atom,+compound,--compound,+list) - one_or_error

Exceptions:
ConnectionOrPool is not a valid reusable connection or pool handle:
domain_error(http_socket_transport_connection_or_pool,ConnectionOrPool)
ConnectionOrPool is not connected to the requested endpoint:
domain_error(http_client_connection_target,endpoint(EndpointHost,
EndpointPort,Host,Port))
URL is a variable:
instantiation_error
URL is not a supported absolute HTTP URL:
domain_error(http_client_url,URL)
URL uses an unsupported HTTP scheme:
domain_error(http_client_scheme,Scheme)
Options contains an invalid HTTP client request option:
domain_error(http_client_request_option,Option)
Body is invalid for the generated normalized HTTP request:
domain_error(http_body,Body)
The delegated socket exchange rejects the response stream:
domain_error(http_response_stream,Error)

---

open_websocket/4

Builds a WebSocket opening-handshake request from the given absolute WebSocket URL and options, selects a compatible transport, opens a reusable socket connection, validates the server `101` response, and returns both the connection handle and the response.

Compilation flags:

`static`

Template:

`open_websocket(URL,Connection,Response,Options)`

Mode and number of proofs:

`open_websocket(+atom,--compound,--compound,+list)` - `one_or_error`

Exceptions:

`URL` is a variable:

`instantiation_error`

`URL` is not a supported absolute WebSocket URL:

`domain_error(http_client_websocket_url,URL)`

`URL` uses an unsupported WebSocket scheme:

`domain_error(http_client_websocket_scheme,Scheme)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid WebSocket client option:

`domain_error(http_client_websocket_option,Option)`

`Options` contains an invalid WebSocket HTTP version:

`domain_error(http_client_websocket_version,Version)`

`Options` contains reserved WebSocket headers:

`domain_error(http_client_websocket_headers,Headers)`

The WebSocket server rejects the version:

`domain_error(http_client_websocket_version_rejection,Response)`

The WebSocket server rejects authentication:

`domain_error(http_client_websocket_authentication_rejection,Response)`

The WebSocket server redirects the opening handshake:

`domain_error(http_client_websocket_redirection_rejection,Response)`

The WebSocket server rejects the opening handshake:

`domain_error(http_client_websocket_rejection,Response)`

The WebSocket server response is not a valid opening handshake response:

`domain_error(http_client_websocket_response,Response)`

open_sse/4

Builds a Server-Sent Events request from the given absolute `http://` or `https://` URL and options, selects a compatible transport, opens a reusable socket connection, writes the request, reads only the response status line and headers, validates the `200 text/event-stream` response, and returns both the connection handle and the response. The response body, if any, is left unread on the connection streams for incremental consumption by the caller.

Compilation flags:

`static`

Template:

`open_sse(URL,Connection,Response,Options)`

Mode and number of proofs:

`open_sse(+atom,--compound,--compound,+list)` - `one_or_error`

Exceptions:

`URL` is a variable:

`instantiation_error`

`URL` is not a supported absolute SSE URL:

`domain_error(http_client_sse_url,URL)`

`URL` uses an unsupported SSE scheme:

`domain_error(http_client_sse_scheme,Scheme)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid SSE client option:

`domain_error(http_client_sse_option,Option)`

`Options` contains reserved SSE headers:

`domain_error(http_client_sse_headers,Headers)`

The SSE server response is not a `200 text/event-stream` response:

`domain_error(http_client_sse_response,Response)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.124 http_client_core

object

1.124.1 `http_client_core`

Portable stream-based HTTP client core predicates.

Availability:
`logtalk_load(http_client_core(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-28

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *http_message_helpers*
Uses:
http_core
list

Remarks:

- Binary streams: All stream predicates expect binary input and output streams.
- Per-message primitives: The `write_request/2`, `read_response/2`, and `exchange/4` predicates process a single HTTP exchange.
- Connection sequence: The `exchange_sequence/4` predicate performs sequential exchanges on the same stream pair while HTTP persistence rules allow it.

Inherited public predicates:
(none)

- *Public predicates*
  - *write_request/2*

  - *read_response/2*
  - *read_response_head/2*
  - *exchange/4*
  - *exchange_sequence/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

write_request/2

Writes exactly one normalized HTTP request term to a binary stream.

Compilation flags:
`static`

Template:
`write_request(Output,Request)`
Mode and number of proofs:
`write_request(+stream,+compound)` - `one_or_error`

Exceptions:
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
The generated request violates normalized HTTP request semantics:
`domain_error(http_header_semantics,Header)`
The generated request properties violate normalized HTTP request semantics:
`domain_error(http_property_semantics,Property)`

---

read_response/2

Reads exactly one HTTP response from a binary stream and returns it as a normalized response term. Fails on clean end-of-file before reading any bytes. Responses may be status-bodyless, Content-Length framed, Transfer-Encoding chunked, or close-delimited by end-of-file. Close-delimited responses are annotated with a `body_framing(close_delimited)` property.

Compilation flags:
`static`

Template:
`read_response(Input,Response)`
Mode and number of proofs:
`read_response(+stream,--compound)` - `zero_or_one`

Exceptions:
The input stream does not contain a valid normalized HTTP response message:

`domain_error(http_response_stream,malformed_response(Bytes))`

`read_response_head/2`

Reads exactly one HTTP response status line and header block from a binary stream and returns a normalized response term with an `empty` body, leaving the stream positioned at the first body byte, if any. Intended for streamed or indeterminate-length response bodies (for example, Server-Sent Events) that callers read incrementally themselves instead of through `read_response/2`.

Compilation flags:

`static`

Template:

`read_response_head(Input,Response)`

Mode and number of proofs:

`read_response_head(+stream,--compound)` - `one_or_error`

Exceptions:

The input stream ends before a status line is read:

`domain_error(http_response_stream,unexpected_end_of_file)`

The input stream does not contain a valid normalized HTTP response status line:

`domain_error(http_response_stream,malformed_status_line(Bytes))`

The input stream contains an invalid response line ending:

`domain_error(http_response_stream,invalid_line_ending(CarriageReturn,LineFeed))`

The input stream contains a bare line feed:

`domain_error(http_response_stream,bare_line_feed)`

`exchange/4`

Writes one normalized request to a binary output stream and then reads one normalized response from a binary input stream.

Compilation flags:

`static`

Template:

`exchange(Input,Output,Request,Response)`

Mode and number of proofs:

`exchange(+stream,+stream,+compound,--compound)` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

The generated request violates normalized HTTP request semantics:

`domain_error(http_header_semantics,Header)`

The generated request properties violate normalized HTTP request semantics:

`domain_error(http_property_semantics,Property)`

The input stream ends before a complete response is read:

`domain_error(http_response_stream,unexpected_end_of_file)`

The input stream does not contain a valid normalized HTTP response status line:

`domain_error(http_response_stream,malformed_status_line(Bytes))`

The input stream contains an unsupported response transfer encoding:

`domain_error(http_response_stream,unsupported_transfer_encoding(Encodings))`

The input stream contains an invalid response line ending:

`domain_error(http_response_stream,invalid_line_ending(CarriageReturn, LineFeed))`

The input stream contains a bare line feed:

`domain_error(http_response_stream,bare_line_feed)`

The input stream does not contain a valid normalized HTTP response message:

`domain_error(http_response_stream,malformed_response(Bytes))`

The response declares an invalid content length:

`domain_error(non_negative_integer,Length)`

---

exchange_sequence/4

Performs a sequence of request-response exchanges on the same binary stream pair. If additional requests remain when HTTP persistence rules require connection closure, an error is thrown.

Compilation flags:

`static`

Template:

`exchange_sequence(Input,Output,Requests,Responses)`

Mode and number of proofs:

`exchange_sequence(+stream,+stream,++list(compound),--list(compound))` - `one_or_error`

Exceptions:

`Requests` is not a valid list of normalized HTTP request terms:

`domain_error(http_client_connection,Requests)`

A request in `Requests` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

HTTP persistence requires the connection to close while requests remain to be exchanged:

`domain_error(http_client_connection,remaining_requests(Requests))`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.125 http_cookies

object

1.125.1 `http_cookies(Representation)`

- `Representation` - Cookie header representation. Valid values are `atom`, `chars`, and `codes`.

HTTP Cookie and `Set-Cookie` header parsing and generation predicates implementing RFC 6265 syntax.

Availability:
`logtalk_load(http_cookies(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-27

Compilation flags:
`static, context_switching_calls`

Uses:
date
list

Remarks:

- Cookie header pairs: Cookie header values are represented as `Name-Value` pairs.
- Set-Cookie components: Set-Cookie header values are represented by a cookie name, a cookie value, and an attribute list where attributes use `Key-Value` notation such as `expires-date_time(Year, Month, Day, Hours, Minutes, Seconds)`, `max_age-Seconds`, `domain-Domain`, `path-Path`, `secure-true`, `http_only-true`, `same_site-lax`, `partitioned-true`, `priority-high`, and `extension-Attribute`.
- Extension attributes: Unknown `Set-Cookie` attributes are preserved verbatim as `extension-Attribute` pairs as long as they do not use a reserved RFC 6265 attribute name.
- Expires attribute: The `Expires` attribute value is normalized to a `date_time(Year, Month, Day, Hours, Minutes, Seconds)` term and generated back using the canonical HTTP-date representation.

Inherited public predicates:
(none)

- *Public predicates*
  - *valid_cookie/1*
  - *parse_cookie/2*
  - *generate_cookie/2*
  - *valid_set_cookie/1*
  - *parse_set_cookie/4*
  - *generate_set_cookie/4*
  - *normalize_cookie_attributes/2*
  - *cookie_attribute_present/2*
  - *cookie_attribute_value/3*
  - *cookie_attribute_value/4*
  - *cookie_expiry/2*
  - *cookie_expiry/3*
  - *cookie_deletion/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

valid_cookie/1

True iff the argument is a valid HTTP `Cookie` header field value.

Compilation flags:
`static`

Template:
`valid_cookie(Cookie)`
Mode and number of proofs:
`valid_cookie(++text)` - `zero_or_one`

---

parse_cookie/2

Parses a `Cookie` header field value into a list of `Name-Value` pairs.

Compilation flags:
`static`

Template:
`parse_cookie(Cookie,Pairs)`
Mode and number of proofs:
`parse_cookie(++text,-list(compound))` - `zero_or_one`

---

generate_cookie/2

Generates a canonical `Cookie` header field value from a list of `Name-Value` pairs using `;` as the separator.

Compilation flags:
`static`

Template:
`generate_cookie(Pairs,Cookie)`
Mode and number of proofs:
`generate_cookie(++list(compound),-text)` - `zero_or_one`

---

valid_set_cookie/1

True iff the argument is a valid HTTP `Set-Cookie` header field value.

Compilation flags:
`static`

Template:
`valid_set_cookie(SetCookie)`
Mode and number of proofs:
`valid_set_cookie(++text)` - `zero_or_one`

---

parse_set_cookie/4

Parses a `Set-Cookie` header field value into a cookie name, a cookie value, and a list of attributes.

Compilation flags:
`static`

Template:
`parse_set_cookie(SetCookie,Name,Value,Attributes)`
Mode and number of proofs:
`parse_set_cookie(++text,-text,-text,-list(compound))` - `zero_or_one`

---

generate_set_cookie/4

Generates a canonical `Set-Cookie` header field value from a cookie name, a cookie value, and a list of attributes.

Compilation flags:
`static`

Template:
`generate_set_cookie(Name,Value,Attributes,SetCookie)`
Mode and number of proofs:
`generate_set_cookie(++text,++text,++list(compound),-text)` - `zero_or_one`

---

normalize_cookie_attributes/2

Normalizes and validates a `Set-Cookie` attribute list using the canonical attribute term representation.

Compilation flags:
`static`

Template:
`normalize_cookie_attributes(Attributes,NormalizedAttributes)`
Mode and number of proofs:
`normalize_cookie_attributes(++list(compound),-list(compound))` - `zero_or_one`

---

cookie_attribute_present/2

True iff a normalized `Set-Cookie` attribute list contains an attribute with the given name.

Compilation flags:
`static`

Template:
`cookie_attribute_present(Attributes,Name)`
Mode and number of proofs:
`cookie_attribute_present(++list(compound),++atom)` - `zero_or_one`

---

cookie_attribute_value/3

Returns the first value for the attribute with the given name in a normalized `Set-Cookie` attribute list.

Compilation flags:
`static`

Template:
`cookie_attribute_value(Attributes,Name,Value)`
Mode and number of proofs:
`cookie_attribute_value(++list(compound),++atom,-term)` - `zero_or_one`

---

cookie_attribute_value/4

Returns the first value for the attribute with the given name in a normalized `Set-Cookie` attribute list or the given default value when absent.

Compilation flags:
`static`

Template:
`cookie_attribute_value(Attributes,Name,Default,Value)`
Mode and number of proofs:
`cookie_attribute_value(++list(compound),++atom,++term,-term)` - `zero_or_one`

---

cookie_expiry/2

Returns the raw normalized expiry view for a `Set-Cookie` attribute list as `session`, `max_age(Seconds)`, or `expires(DateTime)`.

Compilation flags:
`static`

Template:
`cookie_expiry(Attributes,Expiry)`
Mode and number of proofs:
`cookie_expiry(++list(compound),-compound)` - `zero_or_one`

---

cookie_expiry/3

Resolves the normalized expiry view for a `Set-Cookie` attribute list against the given current Unix time as `session`, `delete`, or `expires(UnixTime)`.

Compilation flags:
`static`

Template:
`cookie_expiry(Attributes,CurrentUnixTime,Expiry)`
Mode and number of proofs:
`cookie_expiry(++list(compound),++integer,-compound)` - `zero_or_one`

---

cookie_deletion/3

Returns a canonical `Set-Cookie` term that deletes the named cookie while echoing the relevant scoping and policy attributes from a template attribute list.

Compilation flags:
`static`

Template:
`cookie_deletion(Name,Attributes,Deletion)`
Mode and number of proofs:
`cookie_deletion(++text,++list(compound),-compound)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.126 http_core

category

1.126.1 http_authentication_helpers

Internal shared helpers for HTTP authentication scheme and parameter syntax.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static`

Extends:
`public` *http_text_helpers*
Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *authentication_scheme_codes/3*
  - *parse_authentication_parameters/3*
  - *authentication_token_codes/1*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

authentication_scheme_codes/3

Splits authentication value codes into the scheme and remaining payload codes at the first HTTP optional whitespace code.

Compilation flags:
`static`

Template:
`authentication_scheme_codes(Codes,SchemeCodes,PayloadCodes)`

Mode and number of proofs:
`authentication_scheme_codes(+list(integer),-list(integer),-list(integer))` - `one`

`parse_authentication_parameters/3`

Parses comma-separated authentication parameters, using the given domain-error term for invalid syntax and duplicate names.

Compilation flags:
`static`

Template:
`parse_authentication_parameters(ErrorDomain,Codes,Pairs)`
Mode and number of proofs:
`parse_authentication_parameters(+nonvar,+list(integer),-list(pair))` - `one_or_error`

`authentication_token_codes/1`

Succeeds when the nonempty list of codes is an HTTP token.

Compilation flags:
`static`

Template:
`authentication_token_codes(Codes)`
Mode and number of proofs:
`authentication_token_codes(+list(integer))` - `zero_or_one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.126.2 `http_body_codec_protocol`

Protocol for portable HTTP body codecs.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *media_types/1*
  - *encode_body/4*
  - *decode_body/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

media_types/1

Returns the media types supported by this body codec.

Compilation flags:
`static`

Template:
`media_types(MediaTypes)`
Mode and number of proofs:
`media_types(-list(atom))` - `one`

encode_body/4

Encodes a semantic payload term for the given media type and options into a normalized HTTP body term.

Compilation flags:
`static`

Template:
`encode_body(MediaType,Payload,Options,Body)`
Mode and number of proofs:
`encode_body(+atom,++term,+list(compound),-compound)` - `one_or_error`

Exceptions:
`MediaType` is not a valid HTTP media type atom:
`domain_error(http_media_type,MediaType)`
`Payload` is not valid for `MediaType`:
`domain_error(http_body_payload,Payload)`
`Options` is not a valid HTTP body options list:
`domain_error(http_body_options,Options)`
`Body` is not a valid normalized HTTP body term:
`domain_error(http_body,Body)`

---

decode_body/4

Decodes a normalized HTTP body term for the given media type and options into a semantic payload term.

Compilation flags:
`static`

Template:
`decode_body(MediaType,Body,Options,Payload)`
Mode and number of proofs:
`decode_body(+atom,++compound,+list(compound),-term)` - `one_or_error`

Exceptions:
`MediaType` is not a valid HTTP media type atom:
`domain_error(http_media_type,MediaType)`
`Body` is not a valid normalized HTTP body term:
`domain_error(http_body,Body)`
`Options` is not a valid HTTP body options list:
`domain_error(http_body_options,Options)`
`Payload` cannot be decoded from `Body` for `MediaType`:
`domain_error(http_body_payload,Payload)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.126.3 `http_core`

Transport-independent normalized HTTP request and response constructors, validators, wire parsers and generators, and body codec dispatch.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:4:1
Date: 2026-11-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *http_text_helpers*
Uses:
base64
http_cookies(Representation)
json
list
reader
url(Representation)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *request/7*
  - *response/6*
  - *is_request/1*
  - *is_response/1*

- *parse_request/2*
- *generate_request/2*
- *parse_response/2*
- *generate_response/2*
- *generate_response_headers/2*
- *parse_request_line/2*
- *generate_request_line/2*
- *parse_status_line/2*
- *generate_status_line/2*
- *parse_headers/2*
- *generate_headers/2*
- *parse_body/4*
- *generate_body/3*
- *request_body_bytes/4*
- *response_body_bytes/4*
- *encode_body/4*
- *decode_body/4*
- *method/2*
- *target/2*
- *version/2*
- *status/2*
- *headers/2*
- *header/3*
- *body/2*
- *property/2*

Protected predicates

Private predicates

Operators

Public predicates

request/7

Constructs a validated normalized `request/6` term from the given method, target, version, headers, body, and properties.

Compilation flags:

`static`

Template:

`request(Method,Target,Version,Headers,Body,Properties,Request)`

Mode and number of proofs:

`request(+atom,+compound,+compound,+list(compound),+compound,+list(compound),-compound)` - `one_or_error`

Exceptions:

`Method` is not a valid HTTP method atom:

`domain_error(http_method,Method)`

`Target` is not a valid normalized HTTP target term:

domain_error(http_target,Target)
Version is not a valid HTTP version term:
domain_error(http_version,Version)
Headers is not a valid normalized HTTP header list:
domain_error(http_headers,Headers)
Body is not a valid normalized HTTP body term:
domain_error(http_body,Body)
Properties is not a valid normalized HTTP property list:
domain_error(http_properties,Properties)
The request headers violate normalized HTTP request semantics:
domain_error(http_header_semantics,Header)
The request properties violate normalized HTTP request semantics:
domain_error(http_property_semantics,Property)

---

response/6

Constructs a validated normalized response/5 term from the given version, status, headers, body, and properties.

Compilation flags:
static

Template:
response(Version,Status,Headers,Body,Properties,Response)
Mode and number of proofs:
response(+compound,+compound,+list(compound),+compound,+list(compound),-compound)
- one_or_error

Exceptions:
Version is not a valid HTTP version term:
domain_error(http_version,Version)
Status is not a valid normalized HTTP status term:
domain_error(http_status,Status)
Headers is not a valid normalized HTTP header list:
domain_error(http_headers,Headers)
Body is not a valid normalized HTTP body term:
domain_error(http_body,Body)
Properties is not a valid normalized HTTP property list:
domain_error(http_properties,Properties)
The response headers violate normalized HTTP response semantics:
domain_error(http_header_semantics,Header)
The response properties violate normalized HTTP response semantics:
domain_error(http_property_semantics,Property)

---

is_request/1

Succeeds if the given term is a structurally and semantically valid normalized `request/6` term.

Compilation flags:
`static`

Template:
`is_request(Request)`
Mode and number of proofs:
`is_request(+term)` - `zero_or_one`

---

is_response/1

Succeeds if the given term is a structurally and semantically valid normalized `response/5` term.

Compilation flags:
`static`

Template:
`is_response(Response)`
Mode and number of proofs:
`is_response(+term)` - `zero_or_one`

---

parse_request/2

Parses a complete HTTP request message from the given source (`bytes(List)`, `codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) into a normalized `request/6` term.

Compilation flags:
`static`

Template:
`parse_request(Source,Request)`
Mode and number of proofs:
`parse_request(++compound,--compound)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid HTTP source term:
`domain_error(http_source,Source)`
The parsed request method is not a valid HTTP method atom:
`domain_error(http_method,Method)`
The parsed request target is not a valid normalized HTTP target term:

`domain_error(http_target,Target)`
The parsed request version is not a valid HTTP version term:
`domain_error(http_version,Version)`
The parsed request headers are not a valid normalized HTTP header list:
`domain_error(http_headers,Headers)`
The parsed request body is not a valid normalized HTTP body term:
`domain_error(http_body,Body)`
The parsed request properties are not a valid normalized HTTP property list:
`domain_error(http_properties,Properties)`
The parsed request headers violate normalized HTTP request semantics:
`domain_error(http_header_semantics,Header)`
The parsed request properties violate normalized HTTP request semantics:
`domain_error(http_property_semantics,Property)`

---

generate_request/2

Generates a complete HTTP request message to the given sink (`bytes(List)`, `codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) from a normalized `request/6` term.

Compilation flags:
`static`

Template:
`generate_request(Sink,Request)`
Mode and number of proofs:
`generate_request(+compound,++compound)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid HTTP sink term:
`domain_error(http_sink,Sink)`
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`

---

parse_response/2

Parses a complete HTTP response message from the given source (`bytes(List)`, `codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) into a normalized `response/5` term.

Compilation flags:
`static`

Template:
`parse_response(Source,Response)`

Mode and number of proofs:

`parse_response(++compound,--compound)` - `one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid HTTP source term:

`domain_error(http_source,Source)`

The parsed response version is not a valid HTTP version term:

`domain_error(http_version,Version)`

The parsed response status is not a valid normalized HTTP status term:

`domain_error(http_status,Status)`

The parsed response headers are not a valid normalized HTTP header list:

`domain_error(http_headers,Headers)`

The parsed response body is not a valid normalized HTTP body term:

`domain_error(http_body,Body)`

The parsed response properties are not a valid normalized HTTP property list:

`domain_error(http_properties,Properties)`

The parsed response headers violate normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

The parsed response properties violate normalized HTTP response semantics:

`domain_error(http_property_semantics,Property)`

`generate_response/2`

Generates a complete HTTP response message to the given sink (`bytes(List)`, `codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) from a normalized `response/5` term.

Compilation flags:

`static`

Template:

`generate_response(Sink,Response)`

Mode and number of proofs:

`generate_response(+compound,++compound)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid HTTP sink term:

`domain_error(http_sink,Sink)`

`Response` is not a valid normalized HTTP response term:

`domain_error(http_response,Response)`

---

generate_response_headers/2

Generates an HTTP response status line and effective header block, terminated by an empty line, to the given sink from a normalized `response/5` term.

Compilation flags:

`static`

Template:

`generate_response_headers(Sink,Response)`

Mode and number of proofs:

`generate_response_headers(+compound,++compound)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid HTTP sink term:

`domain_error(http_sink,Sink)`

`Response` is not a valid normalized HTTP response term:

`domain_error(http_response,Response)`

---

parse_request_line/2

Parses an HTTP request line from the given source (`codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) into a `request_line(Method, Target, Version)` term.

Compilation flags:

`static`

Template:

`parse_request_line(Source,RequestLine)`

Mode and number of proofs:

`parse_request_line(++compound,--compound)` - `one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid HTTP source term:

`domain_error(http_source,Source)`

---

generate_request_line/2

Generates an HTTP request line to the given sink (`codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) from a `request_line(Method, Target, Version)` term.

Compilation flags:

`static`

Template:

`generate_request_line(Sink,RequestLine)`

Mode and number of proofs:

`generate_request_line(+compound,++compound)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid HTTP sink term:

`domain_error(http_sink,Sink)`

---

parse_status_line/2

Parses an HTTP status line from the given source (`codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) into a `status_line(Version, Status)` term.

Compilation flags:

`static`

Template:

`parse_status_line(Source,StatusLine)`

Mode and number of proofs:

`parse_status_line(++compound,--compound)` - `one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid HTTP source term:

`domain_error(http_source,Source)`

---

generate_status_line/2

Generates an HTTP status line to the given sink (`codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) from a `status_line(Version, Status)` term.

Compilation flags:

`static`

Template:

`generate_status_line(Sink,StatusLine)`

Mode and number of proofs:

`generate_status_line(+compound,++compound)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid HTTP sink term:

`domain_error(http_sink,Sink)`

---

parse_headers/2

Parses an HTTP header block from the given source (`codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) into normalized header `Name-Value` pairs with typed values for recognized headers.

Compilation flags:

`static`

Template:

`parse_headers(Source,Headers)`

Mode and number of proofs:

`parse_headers(++compound,-list(compound))` - `one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid HTTP source term:

`domain_error(http_source,Source)`

---

generate_headers/2

Generates an HTTP header block to the given sink (`codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) from normalized header `Name-Value` pairs.

Compilation flags:

`static`

Template:

`generate_headers(Sink,Headers)`

Mode and number of proofs:

`generate_headers(+compound,++list(compound))` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid HTTP sink term:

`domain_error(http_sink,Sink)`

`Headers` is not a valid normalized HTTP header list:

`domain_error(http_headers,Headers)`

---

parse_body/4

Parses a body payload from the given source (`bytes(List)`, `codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) according to the given media type and options into a normalized body term.

Compilation flags:

`static`

Template:

`parse_body(Source,MediaType,Options,Body)`

Mode and number of proofs:

`parse_body(++compound,++atom,+list(compound),--compound)` - `one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid HTTP source term:

`domain_error(http_source,Source)`

`MediaType` is not a valid HTTP media type atom:

`domain_error(http_media_type,MediaType)`

`Options` is not a valid HTTP body options list:

`domain_error(http_body_options,Options)`

No registered HTTP body codec exists for `MediaType` when codec-based decoding is required:

`existence_error(http_body_codec,MediaType)`

---

generate_body/3

Generates a body payload to the given sink (`bytes(List)`, `codes(List)`, `chars(List)`, `atom(Atom)`, `file(Path)`, or `stream(Stream)`) from a normalized body term and options.

Compilation flags:

`static`

Template:

`generate_body(Sink,Body,Options)`

Mode and number of proofs:

`generate_body(+compound,++compound,+list(compound))` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid HTTP sink term:

`domain_error(http_sink,Sink)`

`Body` is not a valid normalized HTTP body term:

`domain_error(http_body,Body)`

`Options` is not a valid HTTP body options list:

`domain_error(http_body_options,Options)`

No registered HTTP body codec exists for the body media type when codec-based encoding is required:

`existence_error(http_body_codec,MediaType)`

---

request_body_bytes/4

Generates the transfer-decoded entity-body bytes for a normalized HTTP request body using the request headers and properties to select body serialization options.

Compilation flags:

`static`

Template:

`request_body_bytes(Headers,Body,Properties,Bytes)`

Mode and number of proofs:

`request_body_bytes(+list(compound),++compound,+list(compound),--list(byte))` - `one_or_error`

Exceptions:

`Headers` is not a valid normalized HTTP header list:

`domain_error(http_headers,Headers)`

`Body` is not a valid normalized HTTP body term:

`domain_error(http_body,Body)`

`Properties` is not a valid normalized HTTP property list:

`domain_error(http_properties,Properties)`

---

response_body_bytes/4

Generates the transfer-decoded entity-body bytes for a normalized HTTP response body using the response headers and properties to select body serialization options.

Compilation flags:

`static`

Template:

`response_body_bytes(Headers,Body,Properties,Bytes)`

Mode and number of proofs:

`response_body_bytes(+list(compound),++compound,+list(compound),--list(byte))` - `one_or_error`

Exceptions:

`Headers` is not a valid normalized HTTP header list:

`domain_error(http_headers,Headers)`

`Body` is not a valid normalized HTTP body term:

`domain_error(http_body,Body)`

`Properties` is not a valid normalized HTTP property list:

`domain_error(http_properties,Properties)`

---

encode_body/4

Encodes a semantic payload term using the registered concrete body codec for the given media type into a normalized body term.

Compilation flags:

`static`

Template:

`encode_body(MediaType,Payload,Options,Body)`

Mode and number of proofs:

`encode_body(+atom,++term,+list(compound),-compound)` - `one_or_error`

Exceptions:

`MediaType` is not a valid HTTP media type atom:

`domain_error(http_media_type,MediaType)`

`Options` is not a valid HTTP body options list:

`domain_error(http_body_options,Options)`

No registered HTTP body codec exists for `MediaType`:

`existence_error(http_body_codec,MediaType)`

---

decode_body/4

Decodes a normalized body term using the registered concrete body codec for the given media type into a semantic payload term.

Compilation flags:
`static`

Template:
`decode_body(MediaType,Body,Options,Payload)`
Mode and number of proofs:
`decode_body(+atom,++compound,+list(compound),--term)` - `one_or_error`

Exceptions:
`MediaType` is not a valid HTTP media type atom:
`domain_error(http_media_type,MediaType)`
`Options` is not a valid HTTP body options list:
`domain_error(http_body_options,Options)`
No registered HTTP body codec exists for `MediaType`:
`existence_error(http_body_codec,MediaType)`

---

method/2

Returns the request method from a normalized `request/6` term.

Compilation flags:
`static`

Template:
`method(Request,Method)`
Mode and number of proofs:
`method(+compound,-atom)` - `one_or_error`

Exceptions:

---

target/2

Returns the request target from a normalized `request/6` term.

Compilation flags:
`static`

Template:
`target(Request,Target)`
Mode and number of proofs:
`target(+compound,-compound)` - `one_or_error`

Exceptions:

---

version/2

Returns the HTTP version from a normalized request or response term.

Compilation flags:
```
static
```

Template:
```
version(Message,Version)
```
Mode and number of proofs:
```
version(+compound,-compound) - one_or_error
```

Exceptions:

---

status/2

Returns the response status from a normalized `response/5` term.

Compilation flags:
```
static
```

Template:
```
status(Response,Status)
```
Mode and number of proofs:
```
status(+compound,-compound) - one_or_error
```

Exceptions:

---

headers/2

Returns the normalized header list from a request or response term.

Compilation flags:
```
static
```

Template:
```
headers(Message,Headers)
```
Mode and number of proofs:
```
headers(+compound,-list(compound)) - one_or_error
```

Exceptions:

---

header/3

Enumerates normalized header `Name` and `Value` pairs from a request or response term.

Compilation flags:
`static`

Template:
`header(Message,Name,Value)`
Mode and number of proofs:
`header(+compound,?atom,?term)` - `zero_or_more`

---

body/2

Returns the normalized body term from a request or response term.

Compilation flags:
`static`

Template:
`body(Message,Body)`
Mode and number of proofs:
`body(+compound,-compound)` - `one_or_error`

Exceptions:

---

property/2

Enumerates normalized derived or higher-layer properties from a request or response term.

Compilation flags:
`static`

Template:
`property(Message,Property)`
Mode and number of proofs:
`property(+compound,?compound)` - `zero_or_more`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.126.4 `http_docroot_paths`

Shared helpers for validating requests and sandboxing document-root relative paths.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-05

Compilation flags:
`static`

Uses:
http_core
os

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *validate_relative_path/1*
  - *validate_request/1*
  - *validate_document_root/1*
  - *supported_method/1*
  - *resolved_target_path/3*
  - *path_within_root/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

validate_relative_path/1

Validates a document-root relative request path atom.

Compilation flags:
`static`

Template:
`validate_relative_path(Path)`
Mode and number of proofs:
`validate_relative_path(+atom)` - `one_or_error`

Exceptions:
`Path` is a variable:
`instantiation_error`
`Path` is neither a variable nor an atom:
`type_error(atom,Path)`
`Path` is not a safe document-root relative path:
`domain_error(http_docroot_relative_path,Path)`

---

validate_request/1

Validates a normalized HTTP request term.

Compilation flags:
`static`

Template:
`validate_request(Request)`
Mode and number of proofs:
`validate_request(+compound)` - `one_or_error`

Exceptions:
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`

---

validate_document_root/1

Validates an absolute document root path atom.

Compilation flags:
`static`

Template:
`validate_document_root(DocumentRoot)`
Mode and number of proofs:
`validate_document_root(+atom)` - `one_or_error`

Exceptions:
`DocumentRoot` is a variable:
`instantiation_error`
`DocumentRoot` is neither a variable nor an atom:
`type_error(atom,DocumentRoot)`
`DocumentRoot` is not a valid absolute document root path:
`domain_error(http_docroot_document_root,DocumentRoot)`

---

supported_method/1

True when the request method is supported for document-root serving, currently `get` or `head`.

Compilation flags:
`static`

Template:
`supported_method(Request)`
Mode and number of proofs:
`supported_method(+compound)` - `zero_or_one`

---

resolved_target_path/3

Resolves a request path against a document root and succeeds only when the resulting path stays within that root.

Compilation flags:
`static`

Template:
`resolved_target_path(Path,Root,Candidate)`
Mode and number of proofs:
`resolved_target_path(+atom,+atom,-atom)` - `zero_or_one`

---

path_within_root/2

True when the candidate path is the document root itself or a path nested under it.

Compilation flags:
`static`

Template:
`path_within_root(Root,Candidate)`
Mode and number of proofs:
`path_within_root(+atom,+atom)` - `zero_or_one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.126.5 http_form_body_codec

Concrete HTTP body codec for application/x-www-form-urlencoded payloads.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-23

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_body_codec_protocol*
Uses:
type

Remarks:
(none)

Inherited public predicates:
decode_body/4 *encode_body/4* *media_types/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.126.6 `http_handler_protocol`

Protocol for portable HTTP request handlers.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *handle/2*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`handle/2`

Handles a normalized HTTP request term and returns a normalized HTTP response term.

Compilation flags:
`static`

Template:
`handle(Request,Response)`
Mode and number of proofs:
`handle(+compound,-compound)` - `one_or_error`

Exceptions:
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
`Response` is not a valid normalized HTTP response term:
`domain_error(http_response,Response)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.126.7 `http_json_body_codec`

Concrete HTTP body codec for JSON payloads.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-23

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_body_codec_protocol*

Remarks:
(none)

Inherited public predicates:
decode_body/4 *encode_body/4* *media_types/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.126.8 `http_json_term_helpers`

Internal shared helpers for the repository-standard JSON term representation used by the HTTP, OpenAPI, and HTTP router libraries.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-12

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *json_object_pairs/2*
  - *pairs_to_object/2*
  - *curly_pairs_to_list/2*
  - *list_to_curly_pairs/2*
  - *pair_key_value/3*
  - *normalize_json_value/2*
  - *normalize_json_pairs/2*
- *Private predicates*
- *Operators*

Public predicates

(none)

Protected predicates

json_object_pairs/2

Converts between supported JSON object term representations and a flat list of key-value pairs.

Compilation flags:
`static`

Template:
`json_object_pairs(Object,Pairs)`

Mode and number of proofs:
`json_object_pairs(+term,-list(compound))` - `zero_or_one`
`json_object_pairs(-term,+list(compound))` - `one`

pairs_to_object/2

Converts a list of JSON object key-value pairs into the repository-standard object representation.

Compilation flags:
```
static
```

Template:
```
pairs_to_object(Pairs,Object)
```
Mode and number of proofs:
```
pairs_to_object(+list(compound),-term) - one
```

---

curly_pairs_to_list/2

Converts a nested curly-pair term into a plain list of pairs.

Compilation flags:
```
static
```

Template:
```
curly_pairs_to_list(CurlyPairs,Pairs)
```
Mode and number of proofs:
```
curly_pairs_to_list(+compound,-list(compound)) - one
```

---

list_to_curly_pairs/2

Converts a plain list of pairs into the nested curly-pair representation.

Compilation flags:
```
static
```

Template:
```
list_to_curly_pairs(Pairs,CurlyPairs)
```
Mode and number of proofs:
```
list_to_curly_pairs(+list(compound),-compound) - one
```

---

pair_key_value/3

Normalizes supported key-value pair syntaxes into explicit key and value terms.

Compilation flags:
```
static
```

Template:
```
pair_key_value(Pair,Key,Value)
```
Mode and number of proofs:
```
pair_key_value(+compound,-term,-term) - one
```

---

normalize_json_value/2

Normalizes arbitrary JSON-like values into the repository-standard JSON representation, including booleans and null.

Compilation flags:
```
static
```

Template:
```
normalize_json_value(Value,JsonValue)
```
Mode and number of proofs:
```
normalize_json_value(+term,-term) - one
```

---

normalize_json_pairs/2

Normalizes the values of a JSON object pair list into the repository-standard JSON representation.

Compilation flags:
```
static
```

Template:
```
normalize_json_pairs(Pairs,JsonPairs)
```
Mode and number of proofs:
```
normalize_json_pairs(+list(compound),-list(compound)) - one
```

Private predicates

(none)

Operators

(none)
category

1.126.9 `http_message_helpers`

Internal shared helpers for HTTP transport persistence rules, connection token normalization, and chunk-size line parsing.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-23

Compilation flags:
`static`

Extends:
`public` *http_text_helpers*
Uses:
http_core
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *connection_persistent/2*
  - *request_persistent/1*
  - *response_persistent/1*
  - *version_persistent_by_default/1*
  - *message_has_connection_token/2*
  - *message_connection_tokens/2*
  - *message_header_values/3*
  - *chunk_size_line_size/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

connection_persistent/2

Succeeds when a request-response exchange can continue on the same connection according to HTTP persistence semantics.

Compilation flags:
`static`

Template:
`connection_persistent(Request,Response)`
Mode and number of proofs:
`connection_persistent(+compound,+compound)` - `zero_or_one`

---

request_persistent/1

Succeeds when a normalized HTTP request keeps the connection persistent.

Compilation flags:
`static`

Template:
`request_persistent(Request)`
Mode and number of proofs:
`request_persistent(+compound)` - `zero_or_one`

---

response_persistent/1

Succeeds when a normalized HTTP response keeps the connection persistent.

Compilation flags:
`static`

Template:
`response_persistent(Response)`
Mode and number of proofs:
`response_persistent(+compound)` - `zero_or_one`

---

version_persistent_by_default/1

Succeeds when the given HTTP version uses persistent connections by default.

Compilation flags:

```
static
```

Template:

```
version_persistent_by_default(Version)
```

Mode and number of proofs:

```
version_persistent_by_default(+compound) - zero_or_one
```

---

message_has_connection_token/2

Succeeds when a normalized HTTP message exposes the given normalized Connection token through either headers or derived properties.

Compilation flags:

```
static
```

Template:

```
message_has_connection_token(Message,Token)
```

Mode and number of proofs:

```
message_has_connection_token(+compound,+atom) - zero_or_one
```

---

message_connection_tokens/2

Returns the de-duplicated normalized Connection tokens exposed by a normalized HTTP message.

Compilation flags:

```
static
```

Template:

```
message_connection_tokens(Message,Tokens)
```

Mode and number of proofs:

```
message_connection_tokens(+compound,-list(atom)) - one
```

---

message_header_values/3

Returns the stored header values for the given header name in message order.

Compilation flags:
`static`

Template:
`message_header_values(Message,Name,Values)`
Mode and number of proofs:
`message_header_values(+compound,+atom,-list)` - `one`

---

chunk_size_line_size/2

Parses a chunk-size line into its hexadecimal size, ignoring optional whitespace and chunk extensions.

Compilation flags:
`static`

Template:
`chunk_size_line_size(LineBytes,Size)`
Mode and number of proofs:
`chunk_size_line_size(+list(integer),-integer)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.126.10 http_octet_stream_body_codec

Concrete HTTP body codec for octet-stream payloads.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-23

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_body_codec_protocol*
Uses:
type

Remarks:
(none)

Inherited public predicates:
decode_body/4 *encode_body/4* *media_types/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.126.11 `http_origin_site_helpers`

Internal shared helpers for parsing HTTP origins and absolute URLs, deriving request endpoints, and comparing endpoints for schemeful same-site semantics.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static`

Extends:

`public` *http_text_helpers*

Uses:

atom
http_core
list
url(Representation)
user

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
  - *absolute_url_context/2*
  - *origin_endpoint/2*
  - *request_endpoint/2*
  - *same_site/2*
  - *host_labels/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

absolute_url_context/2

Parses an absolute HTTP or HTTPS URL atom into a normalized endpoint and request path context.

Compilation flags:

`static`

Template:

`absolute_url_context(URL,Context)`

Mode and number of proofs:

`absolute_url_context(+atom,-compound)` - `zero_or_one`

origin_endpoint/2

Parses an Origin header atom into a normalized HTTP or HTTPS endpoint.

Compilation flags:
```
static
```

Template:
```
origin_endpoint(Origin,Endpoint)
```
Mode and number of proofs:
```
origin_endpoint(+atom,-compound) - zero_or_one
```

---

request_endpoint/2

Derives a normalized HTTP or HTTPS endpoint from normalized request properties.

Compilation flags:
```
static
```

Template:
```
request_endpoint(Request,Endpoint)
```
Mode and number of proofs:
```
request_endpoint(+compound,-compound) - zero_or_one
```

---

same_site/2

Succeeds when two normalized endpoints or URL contexts are schemefully same-site using bundled public suffix rules, including exact, wildcard, and exception cases.

Compilation flags:
```
static
```

Template:
```
same_site(Left,Right)
```
Mode and number of proofs:
```
same_site(+compound,+compound) - zero_or_one
```

---

host_labels/2

Splits a normalized host atom into non-empty dot-separated labels.

Compilation flags:
`static`

Template:
`host_labels(Host,Labels)`
Mode and number of proofs:
`host_labels(+atom,-list(atom))` - `zero_or_one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.126.12 http_request_protocol

Protocol for Logtalk objects exposing normalized HTTP request data.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-23

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *method/1*

      - *target/1*
      - *version/1*
      - *headers/1*
      - *header/2*
      - *body/1*
      - *property/1*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

method/1

Returns the normalized lowercase HTTP method atom.

Compilation flags:
`static`

Template:
`method(Method)`
Mode and number of proofs:
`method(-atom) - one`

---

target/1

Returns the normalized request target descriptor.

Compilation flags:
`static`

Template:
`target(Target)`
Mode and number of proofs:
`target(-compound) - one`

---

version/1

Returns the normalized HTTP version term `http(Major, Minor)`.

Compilation flags:
`static`

Template:
`version(Version)`
Mode and number of proofs:
`version(-compound)` - `one`

---

headers/1

Returns the request header list as normalized `Name-Value` pairs.

Compilation flags:
`static`

Template:
`headers(Headers)`
Mode and number of proofs:
`headers(-list(compound))` - `one`

---

header/2

Enumerates normalized request headers as `Name` and `Value` pairs.

Compilation flags:
`static`

Template:
`header(Name,Value)`
Mode and number of proofs:
`header(?atom,?term)` - `zero_or_more`

---

body/1

Returns the normalized request body term.

Compilation flags:
`static`

Template:
`body(Body)`
Mode and number of proofs:
`body(-compound)` - `one`

---

property/1

Enumerates derived or higher-layer normalized request properties.

Compilation flags:
`static`

Template:
`property(Property)`
Mode and number of proofs:
`property(?compound)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.126.13 http_response_protocol

Protocol for Logtalk objects exposing normalized HTTP response data.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-05-23

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *version/1*
  - *status/1*
  - *headers/1*
  - *header/2*
  - *body/1*
  - *property/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

version/1

Returns the normalized HTTP version term `http(Major, Minor)`.

Compilation flags:
`static`

Template:
`version(Version)`

Mode and number of proofs:
`version(-compound)` - `one`

status/1

Returns the normalized HTTP status term `status(Code, ReasonPhrase)`.

Compilation flags:
`static`

Template:
`status(Status)`
Mode and number of proofs:
`status(-compound)` - `one`

---

headers/1

Returns the response header list as normalized `Name-Value` pairs.

Compilation flags:
`static`

Template:
`headers(Headers)`
Mode and number of proofs:
`headers(-list(compound))` - `one`

---

header/2

Enumerates normalized response headers as `Name` and `Value` pairs.

Compilation flags:
`static`

Template:
`header(Name,Value)`
Mode and number of proofs:
`header(?atom,?term)` - `zero_or_more`

---

body/1

Returns the normalized response body term.

Compilation flags:
`static`

Template:
`body(Body)`
Mode and number of proofs:
`body(-compound)` - `one`

---

property/1

Enumerates derived or higher-layer normalized response properties.

Compilation flags:
`static`

Template:
`property(Property)`
Mode and number of proofs:
`property(?compound)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.126.14 http_text_body_codec

Concrete HTTP body codec for text payloads.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-05-23

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_body_codec_protocol*
Uses:
type

Remarks:
(none)

Inherited public predicates:
decode_body/4 *encode_body/4* *media_types/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.126.15 `http_text_helpers`

Internal shared helpers for ASCII case normalization and HTTP optional whitespace trimming.

Availability:
`logtalk_load(http_core(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-05

Compilation flags:
`static`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *ows_code/1*
  - *lowercase_ascii_codes/2*
  - *uppercase_ascii_codes/2*
  - *trim_trailing_ows_codes/2*
  - *trim_ows_codes/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

ows_code/1

Succeeds when the code is HTTP optional whitespace (SP or HTAB).

Compilation flags:
`static`

Template:
`ows_code(Code)`

Mode and number of proofs:
`ows_code(+integer)` - `zero_or_one`

lowercase_ascii_codes/2

Converts uppercase ASCII letter codes in a list to lowercase, leaving all other codes unchanged.

Compilation flags:
```
static
```

Template:
```
lowercase_ascii_codes(Codes,LowercaseCodes)
```
Mode and number of proofs:
```
lowercase_ascii_codes(+list(integer),-list(integer)) - one
```

---

uppercase_ascii_codes/2

Converts lowercase ASCII letter codes in a list to uppercase, leaving all other codes unchanged.

Compilation flags:
```
static
```

Template:
```
uppercase_ascii_codes(Codes,UppercaseCodes)
```
Mode and number of proofs:
```
uppercase_ascii_codes(+list(integer),-list(integer)) - one
```

---

trim_trailing_ows_codes/2

Removes trailing HTTP optional whitespace codes from a list.

Compilation flags:
```
static
```

Template:
```
trim_trailing_ows_codes(Codes,TrimmedCodes)
```
Mode and number of proofs:
```
trim_trailing_ows_codes(+list(integer),-list(integer)) - one
```

---

trim_ows_codes/2

Removes leading and trailing HTTP optional whitespace codes from a list.

Compilation flags:
`static`

Template:
`trim_ows_codes(Codes,TrimmedCodes)`
Mode and number of proofs:
`trim_ows_codes(+list(integer),-list(integer))` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.127 http_cors

object

1.127.1 http_cors

Transport-neutral CORS request classification, preflight response generation, and response decoration helpers for normalized HTTP messages.

Availability:
`logtalk_load(http_cors(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
`public` *http_origin_site_helpers*
Uses:
atom
http_core
list
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *request_origin/2*
  - *is_cors_request/1*
  - *is_preflight_request/1*
  - *preflight_response/3*
  - *add_response_headers/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

request_origin/2

Returns the single request `Origin` header value when present.

Compilation flags:
`static`

Template:
`request_origin(Request,Origin)`
Mode and number of proofs:
`request_origin(+compound,-atom)` - `zero_or_one`

---

is_cors_request/1

Succeeds when the request carries a single `Origin` header value.

Compilation flags:
`static`

Template:
`is_cors_request(Request)`
Mode and number of proofs:
`is_cors_request(+compound)` - `zero_or_one`

---

is_preflight_request/1

Succeeds when the request is a CORS preflight request with method `options` and a single `Access-Control-Request-Method` header value.

Compilation flags:
`static`

Template:
`is_preflight_request(Request)`
Mode and number of proofs:
`is_preflight_request(+compound)` - `zero_or_one`

---

preflight_response/3

Builds a normalized CORS preflight response using the given policy options. Allowed preflight requests return `200 OK` with the relevant `Access-Control-*` headers and denied requests return `403 Forbidden`. Non-preflight requests raise an error.

Compilation flags:
`static`

Template:
`preflight_response(Request,Response,Options)`
Mode and number of proofs:
`preflight_response(+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Request` is not a CORS preflight request:
`domain_error(http_cors_preflight_request,Request)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Options` contains an invalid CORS option combination:
`domain_error(http_cors_options,Options)`
The generated response violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`

---

add_response_headers/4

Decorates a normalized response with the relevant CORS response headers when the request matches and the policy allows it. Denied requests preserve or add cache-relevant `Vary` metadata but do not add permission headers.

Compilation flags:

`static`

Template:

`add_response_headers(Request,Response0,Response,Options)`

Mode and number of proofs:

`add_response_headers(+compound,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid CORS option combination:

`domain_error(http_cors_options,Options)`

`Response0` is not a valid normalized HTTP response term:

`domain_error(http_response,Response0)`

The decorated response violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.128 http_digest

object

1.128.1 `http_client_digest_session`

By deafult, Stateful HTTP Digest client sessions use the `http_socket_transport` library.

Availability:
`logtalk_load(http_digest(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *http_client_digest_session(http_socket_transport)*

Remarks:
(none)

Inherited public predicates:
check_option/1 check_options/1 close/1 cookie_jar/2 default_option/1 default_options/1 delete/4 get/4 head/4 open/3 open/4 option/2 option/3 patch/5 post/5 put/5 request/5 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.128.2 `http_client_digest_session(HTTPTransport)`

Stateful HTTP Digest client sessions that add cookie persistence and one-round-trip Digest challenge retry on top of the normalized HTTP client and socket libraries.

Availability:
`logtalk_load(http_digest(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-12

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
`public` *http_text_helpers*
Uses:
http_cookie_jar
http_cookies(Representation)
http_core
http_digest
http_multipart
list
url(Representation)
user
uuid(Representation)

Remarks:

- Option precedence: When the same session default, Digest default, or per-request option is given multiple times, the first occurrence is used.
- Reactive authentication: Requests are sent once without credentials and retried automatically only when the response carries a `401` Digest challenge accepted by the core `http_digest` object.

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*

    - *open/3*
    - *open/4*
    - *close/1*
    - *cookie_jar/2*
    - *request/5*
    - *get/4*
    - *head/4*
    - *delete/4*
    - *post/5*
    - *put/5*
    - *patch/5*
- *Protected predicates*
- *Private predicates*
    - *session_seed_/1*
    - *session_state_/2*
- *Operators*

Public predicates

open/3

Opens a new Digest client session with the given username and password and a freshly created owned cookie jar.

Compilation flags:
```
static
```

Template:
```
open(Session,Username,Password)
```
Mode and number of proofs:
```
open(-compound,++text,++text) - one_or_error
```

Exceptions:

---

open/4

Opens a new Digest client session using the given credentials plus cookie-jar, request-default, and default Digest authorization options.

Compilation flags:
```
static
```

Template:
```
open(Session,Username,Password,Options)
```
Mode and number of proofs:
```
open(-compound,++text,++text,+list(compound)) - one_or_error
```

Exceptions:

`Username` or `Password` is neither a variable nor text:
```
type_error(text,Text)
```
`Options` is a variable or a partial list:
```
instantiation_error
```
`Options` is neither a variable nor a list:
```
type_error(list,Options)
```
An element `Option` of the list `Options` is neither a variable nor a compound term:
```
type_error(compound,Option)
```
An element `Option` of the list `Options` is a compound term but not a valid option:
```
domain_error(option,Option)
```
`Options` contains an invalid Digest client session option:
```
domain_error(http_client_digest_session_option,Option)
```
`Options` contains an invalid Digest client session option combination:
```
domain_error(http_client_digest_session_options,Options)
```
`Options` contains invalid persisted cookie data:
```
domain_error(http_cookie_jar_persisted_cookies,PersistedCookies)
```

---

close/1

Closes a Digest client session and, when applicable, the owned cookie jar created for it.

Compilation flags:
```
static
```

Template:
```
close(Session)
```
Mode and number of proofs:
```
close(+compound) - one_or_error
```

Exceptions:
`Session` is a variable:
```
instantiation_error
```
`Session` is neither a variable nor an open Digest client session handle:
```
domain_error(http_client_digest_session,Session)
```
`Session` refers to a closed Digest client session handle:
```
existence_error(http_client_digest_session,
http_client_digest_session(SessionId))
```

---

cookie_jar/2

Returns the configured cookie jar handle or the atom `none` when the session does not persist cookies.

Compilation flags:
`static`

Template:
`cookie_jar(Session,Jar)`
Mode and number of proofs:
`cookie_jar(+compound,-term)` - `one_or_error`

Exceptions:
`Session` is a variable:
`instantiation_error`
`Session` is neither a variable nor an open Digest client session handle:
`domain_error(http_client_digest_session,Session)`
`Session` refers to a closed Digest client session handle:
`existence_error(http_client_digest_session,`
`http_client_digest_session(SessionId))`

---

request/5

Performs one HTTP request using session defaults, cookie replay/storage, and automatic retry when the server replies with a Digest challenge.

Compilation flags:
`static`

Template:
`request(Session,Method,URL,Response,Options)`
Mode and number of proofs:
`request(+compound,+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Session` is a variable:
`instantiation_error`
`Session` is neither a variable nor an open Digest client session handle:
`domain_error(http_client_digest_session,Session)`
`Session` refers to a closed Digest client session handle:
`existence_error(http_client_digest_session,`
`http_client_digest_session(SessionId))`
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`URL` uses an unsupported HTTP scheme:
`domain_error(http_client_scheme,Scheme)`

`Options` is a variable or a partial list:
```
instantiation_error
```
`Options` contains invalid Digest client session request options:
```
domain_error(http_client_digest_session_request_options,Options)
```
`Options` contains an invalid Digest client session request option:
```
domain_error(http_client_digest_session_request_option,Option)
```
`Options` contains invalid form-data headers:
```
domain_error(http_client_form_data_headers,Headers)
```
`Options` contains invalid form-data properties:
```
domain_error(http_client_form_data_properties,Properties)
```
The Digest challenge contains an unsupported algorithm:
```
domain_error(http_digest_algorithm,Algorithm)
```
The Digest challenge contains an unsupported qop value:
```
domain_error(http_digest_qop,Qop)
```

---

get/4

Convenience wrapper over `request/5` using the `get` method.

Compilation flags:
```
static
```

Template:
```
get(Session,URL,Response,Options)
```
Mode and number of proofs:
```
get(+compound,+atom,-compound,+list(compound)) - one_or_error
```

Exceptions:
`Session` is not an open Digest client session handle:
```
domain_error(http_client_digest_session,Session)
```
`URL` is not a supported absolute HTTP URL:
```
domain_error(http_client_url,URL)
```
`Options` contains invalid Digest client session request options:
```
domain_error(http_client_digest_session_request_options,Options)
```

---

head/4

Convenience wrapper over `request/5` using the `head` method.

Compilation flags:
```
static
```

Template:
```
head(Session,URL,Response,Options)
```
Mode and number of proofs:

```
head(+compound,+atom,-compound,+list(compound)) - one_or_error
```

Exceptions:

`Session` is not an open Digest client session handle:

```
domain_error(http_client_digest_session,Session)
```

`URL` is not a supported absolute HTTP URL:

```
domain_error(http_client_url,URL)
```

`Options` contains invalid Digest client session request options:

```
domain_error(http_client_digest_session_request_options,Options)
```

---

delete/4

Convenience wrapper over `request/5` using the `delete` method.

Compilation flags:

```
static
```

Template:

```
delete(Session,URL,Response,Options)
```

Mode and number of proofs:

```
delete(+compound,+atom,-compound,+list(compound)) - one_or_error
```

Exceptions:

`Session` is not an open Digest client session handle:

```
domain_error(http_client_digest_session,Session)
```

`URL` is not a supported absolute HTTP URL:

```
domain_error(http_client_url,URL)
```

`Options` contains invalid Digest client session request options:

```
domain_error(http_client_digest_session_request_options,Options)
```

---

post/5

Convenience wrapper over `request/5` using the `post` method and the given body.

Compilation flags:

```
static
```

Template:

```
post(Session,URL,Body,Response,Options)
```

Mode and number of proofs:

```
post(+compound,+atom,+compound,-compound,+list(compound)) - one_or_error
```

Exceptions:

`Session` is not an open Digest client session handle:

domain_error(http_client_digest_session,Session)
URL is not a supported absolute HTTP URL:
domain_error(http_client_url,URL)
Body is invalid for the generated normalized HTTP request:
domain_error(http_body,Body)
Options contains invalid Digest client session request options:
domain_error(http_client_digest_session_request_options,Options)

---

put/5

Convenience wrapper over request/5 using the put method and the given body.

Compilation flags:
static

Template:
put(Session,URL,Body,Response,Options)
Mode and number of proofs:
put(+compound,+atom,+compound,-compound,+list(compound)) - one_or_error

Exceptions:
Session is not an open Digest client session handle:
domain_error(http_client_digest_session,Session)
URL is not a supported absolute HTTP URL:
domain_error(http_client_url,URL)
Body is invalid for the generated normalized HTTP request:
domain_error(http_body,Body)
Options contains invalid Digest client session request options:
domain_error(http_client_digest_session_request_options,Options)

---

patch/5

Convenience wrapper over request/5 using the patch method and the given body.

Compilation flags:
static

Template:
patch(Session,URL,Body,Response,Options)
Mode and number of proofs:
patch(+compound,+atom,+compound,-compound,+list(compound)) - one_or_error

Exceptions:
Session is not an open Digest client session handle:

`domain_error(http_client_digest_session,Session)`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`Body` is invalid for the generated normalized HTTP request:
`domain_error(http_body,Body)`
`Options` contains invalid Digest client session request options:
`domain_error(http_client_digest_session_request_options,Options)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

session_seed_/1

Last allocated Digest client session identifier.

Compilation flags:
`dynamic`

Template:
`session_seed_(SessionId)`
Mode and number of proofs:
`session_seed_(?positive_integer)` - `zero_or_one`

---

session_state_/2

Per-session stored cookie jar ownership, credentials, default request state, and default Digest authorization options.

Compilation flags:
`dynamic`

Template:
`session_state_(SessionId,State)`
Mode and number of proofs:
`session_state_(?positive_integer,?compound)` - `zero_or_more`

Operators

(none)
object

1.128.3 `http_digest`

HTTP Digest authentication parsing, generation, request decoration, and verification helpers.

Availability:
`logtalk_load(http_digest(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2026-10-14

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
`public` *http_authentication_helpers*
Uses:
crypto
date
hmac
http_core
list
os
sha256
type
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *challenge/2*
  - *authorization/2*
  - *authentication_info/2*
  - *parse_challenge/2*
  - *generate_challenge/2*
  - *parse_authorization/2*
  - *generate_authorization/2*
  - *parse_authentication_info/2*

- generate_authentication_info/2
- authorize_request/6
- protect_request/4
- unauthorized_response/3
- unauthorized_response/4
- add_authentication_info/4

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

challenge/2

Returns the single parsed Digest `WWW-Authenticate` challenge from a normalized HTTP response when present.

Compilation flags:
`static`

Template:
`challenge(Response,Challenge)`
Mode and number of proofs:
`challenge(+compound,-compound)` - `zero_or_one`

---

authorization/2

Returns the single parsed Digest `Authorization` header from a normalized HTTP request when present.

Compilation flags:
`static`

Template:
`authorization(Request,Authorization)`
Mode and number of proofs:
`authorization(+compound,-compound)` - `zero_or_one`

---

authentication_info/2

Returns the parsed `Authentication-Info` header from a normalized HTTP response when present.

Compilation flags:

```
static
```

Template:

```
authentication_info(Response,AuthenticationInfo)
```

Mode and number of proofs:

```
authentication_info(+compound,-compound) - zero_or_one
```

---

parse_challenge/2

Parses one Digest challenge header field value into a normalized `digest_challenge/1` term.

Compilation flags:

```
static
```

Template:

```
parse_challenge(Text,Challenge)
```

Mode and number of proofs:

```
parse_challenge(++text,-compound) - one_or_error
```

Exceptions:

`Text` is neither a variable nor text:

```
type_error(text,Text)
```

`Text` is not a valid Digest challenge header value:

```
domain_error(http_digest_header(www_authenticate),invalid(syntax))
```

`Text` uses an unsupported authentication scheme:

```
domain_error(http_digest_header(www_authenticate),
unsupported_scheme(Scheme))
```

`Text` contains a duplicated Digest challenge directive:

```
domain_error(http_digest_header(www_authenticate),duplicate(Name))
```

`Text` contains an unexpected Digest challenge directive:

```
domain_error(http_digest_header(www_authenticate),unexpected(Name))
```

`Text` is missing a required Digest challenge directive:

```
domain_error(http_digest_header(www_authenticate),missing(Name))
```

`Text` contains an invalid Digest challenge directive value:

```
domain_error(http_digest_header(www_authenticate),invalid(Name))
```

---

generate_challenge/2

Generates one canonical Digest challenge header field value from a normalized `digest_challenge/1` term.

Compilation flags:
`static`

Template:
`generate_challenge(Challenge,HeaderValue)`
Mode and number of proofs:
`generate_challenge(+compound,-atom)` - `one_or_error`

Exceptions:
`Challenge` is not a valid normalized Digest challenge term:
`domain_error(http_digest_term(challenge),Challenge)`
`Challenge` is missing a required field:
`domain_error(http_digest_term(challenge),missing(Name))`
`Challenge` contains a duplicated field:
`domain_error(http_digest_term(challenge),duplicate(Name))`
`Challenge` contains an unexpected field:
`domain_error(http_digest_term(challenge),unexpected(Name))`
`Challenge` contains an invalid field:
`domain_error(http_digest_term(challenge),invalid(Name))`
`Challenge` fields are not in canonical order:
`domain_error(http_digest_term(challenge),invalid_order)`
`Challenge` contains an invalid algorithm:
`domain_error(http_digest_algorithm,Algorithm)`
`Challenge` contains an invalid qop value:
`domain_error(http_digest_qop,Qop)`
`Challenge` contains an invalid charset:
`domain_error(http_digest_charset,Charset)`

---

parse_authorization/2

Parses one Digest authorization header field value into a normalized `digest_authorization/1` term.

Compilation flags:
`static`

Template:
`parse_authorization(Text,Authorization)`
Mode and number of proofs:
`parse_authorization(++text,-compound)` - `one_or_error`

Exceptions:
`Text` is neither a variable nor text:
`type_error(text,Text)`

`Text` is not a valid Digest authorization header value:
`domain_error(http_digest_header(authorization),invalid(syntax))`
`Text` uses an unsupported authentication scheme:
`domain_error(http_digest_header(authorization),unsupported_scheme(Scheme))`
`Text` contains a duplicated Digest authorization directive:
`domain_error(http_digest_header(authorization),duplicate(Name))`
`Text` contains an unexpected Digest authorization directive:
`domain_error(http_digest_header(authorization),unexpected(Name))`
`Text` is missing a required Digest authorization directive:
`domain_error(http_digest_header(authorization),missing(Name))`
`Text` contains an invalid Digest authorization directive value:
`domain_error(http_digest_header(authorization),invalid(Name))`

---

generate_authorization/2

Generates one canonical Digest authorization header field value from a normalized `digest_authorization/1` term.

Compilation flags:
`static`

Template:
`generate_authorization(Authorization,HeaderValue)`
Mode and number of proofs:
`generate_authorization(+compound,-atom)` - `one_or_error`

Exceptions:
`Authorization` is not a valid normalized Digest authorization term:
`domain_error(http_digest_term(authorization),Authorization)`
`Authorization` is missing a required field:
`domain_error(http_digest_term(authorization),missing(Name))`
`Authorization` contains a duplicated field:
`domain_error(http_digest_term(authorization),duplicate(Name))`
`Authorization` contains an unexpected field:
`domain_error(http_digest_term(authorization),unexpected(Name))`
`Authorization` contains an invalid field:
`domain_error(http_digest_term(authorization),invalid(Name))`
`Authorization` fields are not in canonical order:
`domain_error(http_digest_term(authorization),invalid_order)`
`Authorization` contains inconsistent qop, nonce-count, and cnonce fields:
`domain_error(http_digest_term(authorization),`
`inconsistent(qop_nonce_count_cnonce))`
`Authorization` contains an invalid algorithm:
`domain_error(http_digest_algorithm,Algorithm)`
`Authorization` contains an invalid qop value:
`domain_error(http_digest_qop,Qop)`

---

parse_authentication_info/2

Parses one `Authentication-Info` header field value into a normalized `digest_authentication_info/1` term.

Compilation flags:
`static`

Template:
`parse_authentication_info(Text,AuthenticationInfo)`
Mode and number of proofs:
`parse_authentication_info(++text,-compound)` - `one_or_error`

Exceptions:
`Text` is neither a variable nor text:
`type_error(text,Text)`
`Text` is not a valid Digest `Authentication-Info` header value:
`domain_error(http_digest_header(authentication_info),invalid(syntax))`
`Text` contains a duplicated Digest authentication-info directive:
`domain_error(http_digest_header(authentication_info),duplicate(Name))`
`Text` contains an unexpected Digest authentication-info directive:
`domain_error(http_digest_header(authentication_info),unexpected(Name))`
`Text` contains an invalid Digest authentication-info directive value:
`domain_error(http_digest_header(authentication_info),invalid(Name))`

---

generate_authentication_info/2

Generates one canonical `Authentication-Info` header field value from a normalized `digest_authentication_info/1` term.

Compilation flags:
`static`

Template:
`generate_authentication_info(AuthenticationInfo,HeaderValue)`
Mode and number of proofs:
`generate_authentication_info(+compound,-atom)` - `one_or_error`

Exceptions:
`AuthenticationInfo` is not a valid normalized Digest authentication-info term:
`domain_error(http_digest_term(authentication_info),AuthenticationInfo)`
`AuthenticationInfo` contains no field to generate:
`domain_error(http_digest_term(authentication_info),missing(all))`
`AuthenticationInfo` contains a duplicated field:
`domain_error(http_digest_term(authentication_info),duplicate(Name))`
`AuthenticationInfo` contains an unexpected field:
`domain_error(http_digest_term(authentication_info),unexpected(Name))`
`AuthenticationInfo` contains an invalid field:

`domain_error(http_digest_term(authentication_info),invalid(Name))`

`AuthenticationInfo` fields are not in canonical order:

`domain_error(http_digest_term(authentication_info),invalid_order)`

`AuthenticationInfo` contains inconsistent qop, nonce-count, and cnonce fields:

`domain_error(http_digest_term(authentication_info),inconsistent(qop_nonce_count_cnonce))`

`authorize_request/6`

Decorates a normalized HTTP request with a Digest `Authorization` header computed from a normalized challenge term, username, password, and options.

Compilation flags:

`static`

Template:

`authorize_request(Request,Challenge,Username,Password,AuthorizedRequest,Options)`

Mode and number of proofs:

`authorize_request(+compound,+compound,++text,++text,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`Challenge` is not a valid normalized Digest challenge term:

`domain_error(http_digest_term(challenge),Challenge)`

`Username` or `Password` is neither a variable nor text:

`type_error(text,Text)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid Digest authorization option:

`domain_error(http_digest_authorize_request_option,Option)`

`Challenge` contains an unsupported algorithm:

`domain_error(http_digest_algorithm,Algorithm)`

`Challenge` contains an unsupported qop value:

`domain_error(http_digest_qop,Qop)`

`Challenge` contains an unsupported charset:

`domain_error(http_digest_charset,Charset)`

The authorized request violates normalized HTTP request semantics:

`domain_error(http_header_semantics,Header)`

---

protect_request/4

Verifies a normalized HTTP request using a Digest verifier object and returns either `continue(Request)` or `respond(Response)`.

Compilation flags:

`static`

Template:

`protect_request(Request,Verifier,Action,Options)`

Mode and number of proofs:

`protect_request(+compound,+object_identifier,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`Verifier` is a variable:

`instantiation_error`

`Verifier` is neither a variable nor an existing object:

`existence_error(http_digest_verifier,Verifier)`

`Verifier` does not implement the Digest verifier protocol:

`domain_error(http_digest_verifier,Verifier)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid Digest protection option:

`domain_error(http_digest_protect_request_option,Option)`

`Options` is missing a required Digest protection option:

`domain_error(http_digest_protect_request_option,Option)`

`Options` contains an invalid Digest status:

`domain_error(http_digest_status,Status)`

`Options` contains an invalid Digest algorithm:

`domain_error(http_digest_algorithm,Algorithm)`

`Options` contains an invalid Digest qop value:

`domain_error(http_digest_qop,Qop)`

`Options` contains an invalid Digest charset:

`domain_error(http_digest_charset,Charset)`

unauthorized_response/3

Builds a normalized `401 Unauthorized` response and returns the generated normalized Digest challenge term.

Compilation flags:

`static`

Template:

`unauthorized_response(Challenge,Response,Options)`

Mode and number of proofs:

`unauthorized_response(-compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid Digest unauthorized-response option:

`domain_error(http_digest_unauthorized_response_option,Option)`

`Options` is missing a required Digest unauthorized-response option:

`domain_error(http_digest_unauthorized_response_option,Option)`

`Options` contains an invalid Digest status:

`domain_error(http_digest_status,Status)`

`Options` contains an invalid Digest algorithm:

`domain_error(http_digest_algorithm,Algorithm)`

`Options` contains an invalid Digest qop value:

`domain_error(http_digest_qop,Qop)`

`Options` contains an invalid Digest charset:

`domain_error(http_digest_charset,Charset)`

The generated response violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

---

unauthorized_response/4

Decorates a normalized HTTP response with an explicit normalized Digest challenge term and returns the resulting `401 Unauthorized` response.

Compilation flags:

`static`

Template:

`unauthorized_response(Challenge,Response0,Response,Options)`

Mode and number of proofs:

`unauthorized_response(+compound,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Challenge` is not a valid normalized Digest challenge term:
`domain_error(http_digest_term(challenge),Challenge)`
`Response0` is not a valid normalized HTTP response term:
`domain_error(http_response,Response0)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Options` contains an invalid Digest unauthorized-response option:
`domain_error(http_digest_unauthorized_response_option,Option)`
`Challenge` contains an invalid Digest charset:
`domain_error(http_digest_charset,Charset)`
`Options` contains an invalid Digest status:
`domain_error(http_digest_status,Status)`
The decorated response violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`

---

add_authentication_info/4

Decorates a normalized HTTP response with an `Authentication-Info` header computed from a previously verified request and options. The `nextnonce` option accepts `false` to omit the field, `true` to generate a fresh nonce using `nonce_secret/1`, or an explicit nonce atom to emit verbatim.

Compilation flags:

`static`

Template:

`add_authentication_info(Request,Response0,Response,Options)`

Mode and number of proofs:

`add_authentication_info(+compound,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
`Request` is not annotated with a verified Digest property:
`domain_error(http_digest_verified_request,missing(Property))`
`Response0` is not a valid normalized HTTP response term:
`domain_error(http_response,Response0)`
`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid authentication-info option:

`domain_error(http_digest_add_authentication_info_option,Option)`

`Options` is missing a required authentication-info option:

`domain_error(http_digest_add_authentication_info_option,Option)`

The decorated response violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.128.4 `http_digest_verifier_protocol`

Protocol for HTTP Digest authentication verifiers backed by stored HA1 values.

Availability:

`logtalk_load(http_digest(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-29

Compilation flags:

`static`

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *ha1/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

ha1/4

Returns the stored lowercase hexadecimal HA1 value for the given normalized digest algorithm family, realm, and username. Fails when the user is unknown for the realm.

Compilation flags:
`static`

Template:
`ha1(Algorithm,Realm,Username,HA1)`

Mode and number of proofs:
`ha1(+atom,++atom,++atom,-atom)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

category

1.128.5 http_router_digest_auth(Verifier,ProtectOptions,AuthenticationInfoOptions)

- `Verifier` - Verifier object passed to `http_digest::protect_request/4` when a routed request declares `digest_auth/1` metadata.
- `ProtectOptions` - Base protect-request options merged with any route-specific `digest_auth/1` metadata options.
- `AuthenticationInfoOptions` - Base options passed to `http_digest::add_authentication_info/4` when a routed request was successfully verified using Digest authentication.

Optional Digest-auth router helpers for objects importing the `http_router` category.

Availability:

```
logtalk_load(http_digest(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-03

Compilation flags:
`static`

Uses:
http_core
http_digest
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *authorize_digest_auth_request/2*
  - *add_digest_authentication_info/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

authorize_digest_auth_request/2

Router helper that applies Digest-auth route metadata to a routed request and returns either `continue(Request)` or `respond(Response)`.

Compilation flags:
`static`

Template:
`authorize_digest_auth_request(Request,Action)`
Mode and number of proofs:
`authorize_digest_auth_request(+compound,-compound)` - `one_or_error`

Exceptions:
`RouteOptions` is not a proper list of Digest protection options:

`domain_error(http_router_digest_auth_options,RouteOptions)`
`ProtectOptions` is not a proper list of Digest protection options:
`domain_error(http_router_digest_auth_options,ProtectOptions)`
`Request` is not a normalized HTTP request term:
`domain_error(http_request,Request)`
`Verifier` is a variable:
`instantiation_error`
`Verifier` does not name an existing object:
`existence_error(http_digest_verifier,Verifier)`
`Verifier` names an object that does not implement `http_digest_verifier_protocol`:
`domain_error(http_digest_verifier,Verifier)`
An element `Option` of the effective Digest protection options list is a variable:
`instantiation_error`
An element `Option` of the effective Digest protection options list is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the effective Digest protection options list is a compound term but not a valid Digest option:
`domain_error(option,Option)`
An element `Option` of the effective Digest protection options list is not accepted by `http_digest::protect_request/4`:
`domain_error(http_digest_protect_request_option,Option)`
The effective Digest protection options omit `realm/1`:
`domain_error(http_digest_term(challenge),missing(realm))`
The effective Digest protection options omit `nonce_secret/1`:
`domain_error(http_digest_protect_request_option,nonce_secret(A))`
The effective Digest protection options request unsupported `userhash` support:
`domain_error(http_digest_term(challenge),invalid(userhash))`
The effective Digest protection options request an unsupported Digest algorithm:
`domain_error(http_digest_algorithm,Algorithm)`
The effective Digest protection options request an unsupported Digest qop:
`domain_error(http_digest_qop,Qop)`
The effective Digest protection options request a non-401 failure status:
`domain_error(http_digest_status,Status)`

---

add_digest_authentication_info/3

Router helper that decorates a response with `Authentication-Info` when the routed request was successfully verified using Digest authentication.

Compilation flags:
`static`

Template:
`add_digest_authentication_info(Request,Response0,Response)`
Mode and number of proofs:
`add_digest_authentication_info(+compound,+compound,-compound)` - `one_or_error`

Exceptions:

`AuthenticationInfoOptions` is not a proper list of Digest authentication-info options:
```
domain_error(http_router_digest_auth_authentication_info_options,
AuthenticationInfoOptions)
```
`Request` is not a normalized HTTP request term:
```
domain_error(http_request,Request)
```
`Response0` is not a normalized HTTP response term:
```
domain_error(http_response,Response0)
```
An element `Option` of the Digest authentication-info options list is a variable:
```
instantiation_error
```
An element `Option` of the Digest authentication-info options list is neither a variable nor a compound term:
```
type_error(compound,Option)
```
An element `Option` of the Digest authentication-info options list is a compound term but not a valid Digest option:
```
domain_error(option,Option)
```
An element `Option` of the Digest authentication-info options list is not accepted by `http_digest::add_authentication_info/4`:
```
domain_error(http_digest_add_authentication_info_option,Option)
```
The routed request carries an invalid Digest authorization term:
```
domain_error(http_digest_term(authorization),Cause)
```
The routed request carries a Digest authorization with an unsupported response qop:
```
domain_error(http_digest_qop,Qop)
```
The routed request carries a Digest authorization with an unsupported algorithm:
```
domain_error(http_digest_algorithm,Algorithm)
```
Automatic `nextnonce` generation is requested without `nonce_secret/1`:
```
domain_error(http_digest_add_authentication_info_option,nonce_secret(A))
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.128.6 `http_server_core_digest_handler(Verifier,Handler,ProtectOptions, AuthenticationInfoOptions)`

- `Verifier` - Verifier object implementing the `http_digest_verifier_protocol` protocol.
- `Handler` - Wrapped object implementing the `http_handler_protocol` protocol.
- `ProtectOptions` - Options passed to `http_digest::protect_request/4`.
- `AuthenticationInfoOptions` - Options passed to `http_digest::add_authentication_info/4` on successful request verification.

Wrapper HTTP handler object that applies Digest request verification and Authentication-Info response decoration around another portable HTTP handler.

Availability:
```
logtalk_load(http_digest(loader))
```

Author: Paulo Moura

Version: 1:0:0
Date: 2026-07-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_handler_protocol*
Uses:
http_digest

Remarks:
(none)

Inherited public predicates:
handle/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.129 http_directory_listing

object

1.129.1 http_directory_listing

Router-agnostic directory listing helper built on the normalized `http_core` library.

Availability:
`logtalk_load(http_directory_listing(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
`public` *http_docroot_paths*

Uses:
atom
date
html5
http_core
list
mime_types
os
term_io
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *serve/4*
  - *serve/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

serve/4

Serves a directory listing for a relative request path from the given document root using the default options and returns a normalized response.

Compilation flags:

`static`

Template:

`serve(Path,Request,DocumentRoot,Response)`

Mode and number of proofs:

`serve(+atom,+compound,+atom,-compound)` - `one_or_error`

Exceptions:

`Path` is not a safe document-root relative path:

`domain_error(http_docroot_relative_path,Path)`

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`DocumentRoot` is not a valid absolute document root path:

`domain_error(http_docroot_document_root,DocumentRoot)`

`DocumentRoot` is not an existing directory-listing document root:

`domain_error(http_directory_listing_document_root,DocumentRoot)`

The generated response violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

---

serve/5

Serves a directory listing for a relative request path from the given document root using the given options and returns a normalized response.

Compilation flags:

`static`

Template:

`serve(Path,Request,DocumentRoot,Response,Options)`

Mode and number of proofs:

`serve(+atom,+compound,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Path` is not a safe document-root relative path:

`domain_error(http_docroot_relative_path,Path)`

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`DocumentRoot` is not a valid absolute document root path:

`domain_error(http_docroot_document_root,DocumentRoot)`

`DocumentRoot` is not an existing directory-listing document root:

`domain_error(http_directory_listing_document_root,DocumentRoot)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The generated response violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.130 http_htmx

object

1.130.1 `http_htmx`

Transport-neutral HTMX request classification, HTML reply helpers, and response decoration helpers for normalized HTTP messages.

Availability:

`logtalk_load(http_htmx(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-07

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *options*
`public` *http_json_term_helpers*

Uses:

atom

html5
http_core
list
term_io
url(Representation)
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - `is_htmx_request/1`
  - `is_boosted_request/1`
  - `is_history_restore_request/1`
  - `is_fragment_request/1`
  - `request_kind/2`
  - `current_url/2`
  - `current_url_abs_path/2`
  - `prompt/2`
  - `target/2`
  - `trigger/2`
  - `trigger_name/2`
  - `request_properties/2`
  - `request_property/2`
  - `reply/3`
  - `reply/4`
  - `page_fragment_reply/4`
  - `page_fragment_reply/5`
  - `add_response_headers/4`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`is_htmx_request/1`

Succeeds when the request carries a truthy `HX-Request` header value.

Compilation flags:
`static`

Template:
`is_htmx_request(Request)`

Mode and number of proofs:

`is_htmx_request(+compound)` - `zero_or_one`

---

is_boosted_request/1

Succeeds when the request carries a truthy `HX-Boosted` header value.

Compilation flags:

`static`

Template:

`is_boosted_request(Request)`

Mode and number of proofs:

`is_boosted_request(+compound)` - `zero_or_one`

---

is_history_restore_request/1

Succeeds when the request carries a truthy `HX-History-Restore-Request` header value.

Compilation flags:

`static`

Template:

`is_history_restore_request(Request)`

Mode and number of proofs:

`is_history_restore_request(+compound)` - `zero_or_one`

---

is_fragment_request/1

Succeeds when the request should receive fragment content, meaning an HTMX request that is neither boosted nor a history restore request.

Compilation flags:

`static`

Template:

`is_fragment_request(Request)`

Mode and number of proofs:

`is_fragment_request(+compound)` - `zero_or_one`

---

request_kind/2

Classifies the request as one of `ordinary`, `fragment`, `boosted`, or `history_restore`.

Compilation flags:
`static`

Template:
`request_kind(Request,Kind)`
Mode and number of proofs:
`request_kind(+compound,-atom)` - `one_or_error`

Exceptions:

---

current_url/2

Returns the single request `HX-Current-URL` header value when present.

Compilation flags:
`static`

Template:
`current_url(Request,URL)`
Mode and number of proofs:
`current_url(+compound,-atom)` - `zero_or_one`

---

current_url_abs_path/2

Returns the same-origin absolute-path form of the `HX-Current-URL` header value, including any query component but excluding scheme, authority, and fragment.

Compilation flags:
`static`

Template:
`current_url_abs_path(Request,AbsolutePath)`
Mode and number of proofs:
`current_url_abs_path(+compound,-atom)` - `zero_or_one`

---

prompt/2

Returns the single request `HX-Prompt` header value when present.

Compilation flags:
`static`

Template:
`prompt(Request,Prompt)`
Mode and number of proofs:
`prompt(+compound,-atom)` - `zero_or_one`

---

target/2

Returns the single request `HX-Target` header value when present.

Compilation flags:
`static`

Template:
`target(Request,Target)`
Mode and number of proofs:
`target(+compound,-atom)` - `zero_or_one`

---

trigger/2

Returns the single request `HX-Trigger` header value when present.

Compilation flags:
`static`

Template:
`trigger(Request,Trigger)`
Mode and number of proofs:
`trigger(+compound,-atom)` - `zero_or_one`

---

trigger_name/2

Returns the single request `HX-Trigger-Name` header value when present.

Compilation flags:
`static`

Template:
`trigger_name(Request,Name)`
Mode and number of proofs:
`trigger_name(+compound,-atom)` - `zero_or_one`

---

request_properties/2

Returns a list of derived HTMX request properties and request classification computed from the normalized request headers.

Compilation flags:
`static`

Template:
`request_properties(Request,Properties)`
Mode and number of proofs:
`request_properties(+compound,-list(compound))` - `one_or_error`

Exceptions:

---

request_property/2

Enumerates or tests individual derived HTMX request properties and request classification computed from the normalized request headers.

Compilation flags:
`static`

Template:
`request_property(Request,Property)`
Mode and number of proofs:
`request_property(+compound,?compound)` - `zero_or_more`

---

reply/3

Builds a normalized `text/html` response from a pre-rendered HTML atom or an `html` term/list using the default options.

Compilation flags:

`static`

Template:

`reply(Request,Content,Response)`

Mode and number of proofs:

`reply(+compound,+term,-compound)` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`Content` cannot be rendered as HTMX HTML content:

`domain_error(http_htmx_content,Content)`

The HTMX response status is invalid:

`domain_error(htmx_response_headers_status,Status)`

The generated response violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

---

reply/4

Builds a normalized `text/html` response from a pre-rendered HTML atom or an `html` term/list using the given options.

Compilation flags:

`static`

Template:

`reply(Request,Content,Response,Options)`

Mode and number of proofs:

`reply(+compound,+term,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`Content` cannot be rendered as HTMX HTML content:

`domain_error(http_htmx_content,Content)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The HTMX response status is invalid:
`domain_error(htmx_response_headers_status,Status)`
The generated response violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`

---

page_fragment_reply/4

Builds a normalized `text/html` response using the fragment content when `is_fragment_request/1` succeeds and the full-page content otherwise, with the default options.

Compilation flags:
`static`

Template:
`page_fragment_reply(Request,PageContent,FragmentContent,Response)`
Mode and number of proofs:
`page_fragment_reply(+compound,+term,+term,-compound)` - `one_or_error`

Exceptions:
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
`PageContent` or `FragmentContent` cannot be rendered as HTMX HTML content:
`domain_error(http_htmx_content,Content)`
The HTMX response status is invalid:
`domain_error(htmx_response_headers_status,Status)`
The generated response violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`

---

page_fragment_reply/5

Builds a normalized `text/html` response using the fragment content when `is_fragment_request/1` succeeds and the full-page content otherwise, with the given options.

Compilation flags:
`static`

Template:
`page_fragment_reply(Request,PageContent,FragmentContent,Response,Options)`
Mode and number of proofs:
`page_fragment_reply(+compound,+term,+term,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`

`PageContent` or `FragmentContent` cannot be rendered as HTMX HTML content:
`domain_error(http_htmx_content,Content)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The HTMX response status is invalid:
`domain_error(htmx_response_headers_status,Status)`
The generated response violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`

---

add_response_headers/4

Decorates a normalized response with the relevant HTMX response headers and optional cache-vary metadata.

Compilation flags:
`static`

Template:
`add_response_headers(Request,Response0,Response,Options)`
Mode and number of proofs:
`add_response_headers(+compound,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
`Response0` is not a valid normalized HTTP response term:
`domain_error(http_response,Response0)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The HTMX response status is invalid:
`domain_error(htmx_response_headers_status,Status)`
The decorated response violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.130.2 `http_router_htmx`

Optional HTMX middleware helpers for router objects importing the `http_router` category.

Availability:
`logtalk_load(http_htmx(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static`

Uses:
http_core
http_htmx
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *annotate_htmx_request/2*
  - *add_htmx_response_headers/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

annotate_htmx_request/2

Router middleware helper that annotates a normalized request with derived HTMX properties and continues the middleware chain.

Compilation flags:

`static`

Template:

`annotate_htmx_request(Request,Action)`

Mode and number of proofs:

`annotate_htmx_request(+compound,-compound)` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

The annotated request violates normalized HTTP request semantics:

`domain_error(http_header_semantics,Header)`

The annotated request properties violate normalized HTTP request semantics:

`domain_error(http_property_semantics,Property)`

---

add_htmx_response_headers/3

Router response-middleware helper that decorates a normalized response using `htmx_response_options/1` request or response properties.

Compilation flags:

`static`

Template:

`add_htmx_response_headers(Request,Response0,Response)`

Mode and number of proofs:

`add_htmx_response_headers(+compound,+compound,-compound)` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`Response0` is not a valid normalized HTTP response term:

`domain_error(http_response,Response0)`

The derived HTMX response options are invalid:

`domain_error(option,Option)`

The decorated response violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.131 http_multipart

object

1.131.1 `http_multipart`

Multipart helper predicates built on top of the normalized body and part terms provided by the http library.

Availability:
`logtalk_load(http_multipart(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *http_text_helpers*
Uses:
http_core
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *body/3*
  - *is_body/1*
  - *parse/4*
  - *generate/3*
  - *media_type/2*
  - *parts/2*
  - *fields/2*
  - *files/2*

      - *part/4*
      - *is_part/1*
      - *part_headers/2*
      - *part_body/2*
      - *part_properties/2*
      - *field/4*
      - *file/6*
      - *field_part/4*
      - *file_part/6*
      - *form_data_body/2*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

body/3

Constructs a validated normalized multipart body term from a multipart media type atom and a list of multipart parts.

Compilation flags:
`static`

Template:
`body(MediaType,Parts,Body)`
Mode and number of proofs:
`body(+atom,+list(compound),-compound)` - `one_or_error`

Exceptions:
`MediaType` is not a valid multipart media type atom:
`domain_error(http_multipart_media_type,MediaType)`
`Parts` is not a valid list of normalized multipart parts:
`domain_error(http_multipart_parts,Parts)`

---

is_body/1

True when the argument is a valid normalized multipart body term.

Compilation flags:
`static`

Template:
`is_body(Body)`
Mode and number of proofs:
`is_body(@term)` - `zero_or_one`

---

parse/4

Parses a multipart body from a source term by delegating to the underlying `http_core::parse_body/4` predicate after validating the media type.

Compilation flags:
`static`

Template:
`parse(Source,MediaType,Options,Body)`
Mode and number of proofs:
`parse(++compound,+atom,+list,-compound)` - `one_or_error`

Exceptions:
`MediaType` is not a valid multipart media type atom:
`domain_error(http_multipart_media_type,MediaType)`
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid HTTP source term:
`domain_error(http_source,Source)`
`Options` is not a valid HTTP body options list:
`domain_error(http_body_options,Options)`
No registered HTTP body codec exists for `MediaType` when codec-based decoding is required:
`existence_error(http_body_codec,MediaType)`
The parsed body is not a valid normalized multipart body term:
`domain_error(http_multipart_body,Body)`

---

generate/3

Generates a multipart body to a sink term by delegating to the underlying `http_core::generate_body/3` predicate after validating the multipart body term.

Compilation flags:
`static`

Template:
`generate(Sink,Body,Options)`
Mode and number of proofs:
`generate(++compound,+compound,+list)` - `one_or_error`

Exceptions:
`Body` is not a valid normalized multipart body term:
`domain_error(http_multipart_body,Body)`
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid HTTP sink term:
`domain_error(http_sink,Sink)`

`Options` is not a valid HTTP body options list:
`domain_error(http_body_options,Options)`
No registered HTTP body codec exists for the body media type when codec-based encoding is required:
`existence_error(http_body_codec,MediaType)`

---

media_type/2

Returns the media type atom of a validated multipart body term.

Compilation flags:
`static`

Template:
`media_type(Body,MediaType)`
Mode and number of proofs:
`media_type(+compound,-atom)` - `one_or_error`

Exceptions:
`Body` is not a valid normalized multipart body term:
`domain_error(http_multipart_body,Body)`

---

parts/2

Returns the list of multipart part terms of a validated multipart body term.

Compilation flags:
`static`

Template:
`parts(Body,Parts)`
Mode and number of proofs:
`parts(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Body` is not a valid normalized multipart body term:
`domain_error(http_multipart_body,Body)`

---

fields/2

Returns the list of form-data field(Name, Value, Parameters) descriptors found in a validated multipart/form-data body, preserving part order.

Compilation flags:
`static`

Template:
`fields(Body,Fields)`
Mode and number of proofs:
`fields(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Body` is not a valid normalized multipart/form-data body term:
`domain_error(http_multipart_form_data_body,Body)`

---

files/2

Returns the list of form-data `file(Name, Filename, MediaType, Payload, Parameters)` descriptors found in a validated multipart/form-data body, preserving part order.

Compilation flags:
`static`

Template:
`files(Body,Files)`
Mode and number of proofs:
`files(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Body` is not a valid normalized multipart/form-data body term:
`domain_error(http_multipart_form_data_body,Body)`

---

part/4

Constructs a validated normalized multipart part term from headers, body, and properties.

Compilation flags:
`static`

Template:
`part(Headers,Body,Properties,Part)`
Mode and number of proofs:
`part(+list(compound),+compound,+list(compound),-compound)` - `one_or_error`

Exceptions:
The given headers, body, and properties do not form a valid normalized multipart part term:
`domain_error(http_multipart_part,Part)`

---

is_part/1

True when the argument is a valid normalized multipart part term.

Compilation flags:
`static`

Template:
`is_part(Part)`
Mode and number of proofs:
`is_part(@term)` - `zero_or_one`

---

part_headers/2

Returns the headers of a validated multipart part term.

Compilation flags:
`static`

Template:
`part_headers(Part,Headers)`
Mode and number of proofs:
`part_headers(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Part` is not a valid normalized multipart part term:
`domain_error(http_multipart_part,Part)`

---

part_body/2

Returns the body of a validated multipart part term.

Compilation flags:
`static`

Template:
`part_body(Part,Body)`
Mode and number of proofs:

`part_body(+compound,-compound)` - `one_or_error`

Exceptions:

`Part` is not a valid normalized multipart part term:

`domain_error(http_multipart_part,Part)`

---

part_properties/2

Returns the properties of a validated multipart part term.

Compilation flags:

`static`

Template:

`part_properties(Part,Properties)`

Mode and number of proofs:

`part_properties(+compound,-list(compound))` - `one_or_error`

Exceptions:

`Part` is not a valid normalized multipart part term:

`domain_error(http_multipart_part,Part)`

---

field/4

True when the validated multipart part is a textual form-data field part, returning its field name, text value, and additional disposition parameters.

Compilation flags:

`static`

Template:

`field(Part,Name,Value,Parameters)`

Mode and number of proofs:

`field(+compound,-atom,-term,-list(compound))` - `zero_or_one`

---

file/6

True when the validated multipart part is a form-data file part, returning its field name, filename, media type, payload term, and additional disposition parameters.

Compilation flags:

`static`

Template:

`file(Part,Name,Filename,MediaType,Payload,Parameters)`

Mode and number of proofs:

`file(+compound,-atom,-atom,-atom,-compound,-list(compound))` - `zero_or_one`

---

field_part/4

Constructs a normalized multipart part for a textual form-data field using a content-disposition header, a text/plain body, and additional disposition parameters.

Compilation flags:

`static`

Template:

`field_part(Name,Value,Parameters,Part)`

Mode and number of proofs:

`field_part(+atom,+term,+list(compound),-compound)` - `one_or_error`

Exceptions:

`Name` is not a valid form-data field name:

`domain_error(http_multipart_form_data_parameter_name,Name)`

`Parameters` is not a valid list of additional form-data disposition parameters:

`domain_error(http_multipart_form_data_parameters,Parameters)`

A form-data disposition parameter is invalid for a textual field part:

`domain_error(http_multipart_form_data_parameter,Parameter)`

---

file_part/6

Constructs a normalized multipart part for a form-data file using a content-disposition header, the given media type and payload, and additional disposition parameters.

Compilation flags:

`static`

Template:

`file_part(Name,Filename,MediaType,Payload,Parameters,Part)`

Mode and number of proofs:

`file_part(+atom,+atom,+atom,+compound,+list(compound),-compound)` - `one_or_error`

Exceptions:

`Name` is not a valid form-data field name:

`domain_error(http_multipart_form_data_parameter_name,Name)`

`Filename` is not a valid form-data filename value:

`domain_error(http_multipart_form_data_parameter_value,Filename)`

`Parameters` is not a valid list of additional form-data disposition parameters:

`domain_error(http_multipart_form_data_parameters,Parameters)`
A form-data disposition parameter is invalid for a file part:
`domain_error(http_multipart_form_data_parameter,Parameter)`

`form_data_body/2`

Constructs a multipart/form-data body from an ordered list of `field(Name, Value, Parameters)` and `file(Name, Filename, MediaType, Payload, Parameters)` descriptors.

Compilation flags:
`static`

Template:
`form_data_body(Items,Body)`
Mode and number of proofs:
`form_data_body(+list(compound),-compound)` - `one_or_error`

Exceptions:
`Items` is not a valid list of form-data field or file descriptors:
`domain_error(http_multipart_form_data_items,Items)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.132 http_oauth

object

1.132.1 `http_oauth`

OAuth Bearer parsing, challenge, authentication, authorization, and response helpers.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-09-01

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
`public` *http_oauth_helpers*

Uses:
http_core
list
type
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *challenge/2*
  - *authorization/2*
  - *parse_challenge/2*
  - *generate_challenge/2*
  - *parse_authorization/2*
  - *generate_authorization/2*
  - *authenticate_request/4*
  - *authorize_request/3*
  - *protect_request/4*
  - *token_info/2*
  - *scopes/2*
  - *unauthorized_response/3*
  - *unauthorized_response/4*
  - *forbidden_response/3*
  - *forbidden_response/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

challenge/2

Returns the single parsed Bearer challenge from a normalized HTTP response. Fails when the response contains no Bearer `WWW-Authenticate` header field.

Compilation flags:
`static`

Template:
`challenge(Response,Challenge)`
Mode and number of proofs:
`challenge(+compound,-compound)` - `zero_or_one_or_error`

Exceptions:
`Response` is not a valid normalized HTTP response term:
`domain_error(http_response,Response)`
`Response` contains multiple Bearer `WWW-Authenticate` header values `Values`:
`domain_error(http_oauth_header_values(www_authenticate),Values)`
A Bearer `WWW-Authenticate` header field uses unsupported authentication scheme `Scheme`:
`domain_error(http_oauth_bearer_scheme,Scheme)`
A Bearer `WWW-Authenticate` header field value `Value` has invalid syntax:
`domain_error(http_oauth_header(www_authenticate),Value)`
A Bearer `WWW-Authenticate` header field value `Value` contains invalid or duplicate parameters:
`domain_error(http_oauth_challenge_parameters,Value)`
A Bearer `WWW-Authenticate` header field contains invalid scope value `Scope`:
`domain_error(http_oauth_scope,Scope)`
A Bearer `WWW-Authenticate` header field contains invalid parameter name `Name`:
`domain_error(http_authentication_parameter_name,Name)`

---

authorization/2

Returns the single parsed Bearer authorization from a normalized HTTP request. Fails when the request contains no Bearer `Authorization` header field.

Compilation flags:
`static`

Template:
`authorization(Request,Authorization)`
Mode and number of proofs:
`authorization(+compound,-compound)` - `zero_or_one_or_error`

Exceptions:
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
`Request` contains multiple Bearer `Authorization` header values `Values`:

`domain_error(http_oauth_header_values(authorization),Values)`
A Bearer `Authorization` header field uses unsupported authentication scheme `Scheme`:
`domain_error(http_oauth_bearer_scheme,Scheme)`
A Bearer `Authorization` header field value `Value` has invalid syntax:
`domain_error(http_oauth_header(authorization),Value)`
A Bearer `Authorization` header field contains invalid token `Token`:
`domain_error(http_oauth_bearer_token,Token)`

---

parse_challenge/2

Parses a Bearer challenge header value.

Compilation flags:
`static`

Template:
`parse_challenge(Text,Challenge)`
Mode and number of proofs:
`parse_challenge(++text,-compound)` - `one_or_error`

Exceptions:
`Text` is a variable:
`instantiation_error`
`Text` is neither a variable nor text:
`type_error(text,Text)`
`Text` uses unsupported authentication scheme `Scheme`:
`domain_error(http_oauth_bearer_scheme,Scheme)`
`Text` is not a valid Bearer challenge header value:
`domain_error(http_oauth_header(www_authenticate),Text)`
`Text` contains invalid or duplicate Bearer challenge parameters:
`domain_error(http_oauth_challenge_parameters,Text)`
`Text` contains invalid Bearer scope value `Scope`:
`domain_error(http_oauth_scope,Scope)`
`Text` contains invalid Bearer challenge parameter name `Name`:
`domain_error(http_authentication_parameter_name,Name)`

---

generate_challenge/2

Generates a Bearer challenge header value.

Compilation flags:
`static`

Template:
`generate_challenge(Challenge,HeaderValue)`

Mode and number of proofs:

`generate_challenge(+compound,-atom)` - `one_or_error`

Exceptions:

`Challenge` is not a valid normalized Bearer challenge term or contains duplicate fields:

`domain_error(http_oauth_term(challenge),Challenge)`

---

parse_authorization/2

Parses a Bearer authorization header value.

Compilation flags:

`static`

Template:

`parse_authorization(Text,Authorization)`

Mode and number of proofs:

`parse_authorization(++text,-compound)` - `one_or_error`

Exceptions:

`Text` is a variable:

`instantiation_error`

`Text` is neither a variable nor text:

`type_error(text,Text)`

`Text` uses unsupported authentication scheme `Scheme`:

`domain_error(http_oauth_bearer_scheme,Scheme)`

`Text` is not a valid Bearer authorization header value:

`domain_error(http_oauth_header(authorization),Text)`

`Text` contains invalid Bearer token `Token`:

`domain_error(http_oauth_bearer_token,Token)`

---

generate_authorization/2

Generates a Bearer authorization header value.

Compilation flags:

`static`

Template:

`generate_authorization(Authorization,HeaderValue)`

Mode and number of proofs:

`generate_authorization(+compound,-atom)` - `one_or_error`

Exceptions:

`Authorization` is not a valid normalized Bearer authorization term:
`domain_error(http_oauth_term(authorization),Authorization)`

`authenticate_request/4`

Authenticates a normalized request, returning continue(AnnotatedRequest) or respond(Response).

Compilation flags:
`static`

Template:
`authenticate_request(Request,Verifier,Action,Options)`

Mode and number of proofs:
`authenticate_request(+compound,+object_identifier,--compound,+list(compound))` - `one_or_error`

Exceptions:
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
`Verifier` is a variable:
`instantiation_error`
`Verifier` does not name an existing object:
`existence_error(http_oauth_verifier,Verifier)`
`Verifier` does not implement `http_oauth_verifier_protocol`:
`domain_error(http_oauth_verifier,Verifier)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The scope checker `ScopeChecker` selected by `Options` does not name an existing object:
`existence_error(http_oauth_scope_checker,ScopeChecker)`
The scope checker `ScopeChecker` selected by `Options` does not implement `http_oauth_scope_checker_protocol`:
`domain_error(http_oauth_scope_checker,ScopeChecker)`
`Options` omits a protected-resource identifier:
`domain_error(http_oauth_protection_options,Options)`
`Verifier` returns malformed normalized token information `TokenInfo`:
`domain_error(http_oauth_token_info,TokenInfo)`
The generated response properties `Properties` are invalid:
`domain_error(http_properties,Properties)`
The generated response header `Header` violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`
The generated response property `Property` violates normalized HTTP response semantics:
`domain_error(http_property_semantics,Property)`

`authorize_request/3`

Authorizes an authenticated request against required scopes, returning continue(Request) or respond(Response).

Compilation flags:
`static`

Template:
`authorize_request(Request,Action,Options)`
Mode and number of proofs:
`authorize_request(+compound,--compound,+list(compound))` - `one_or_error`

Exceptions:
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The scope checker `ScopeChecker` selected by `Options` does not name an existing object:
`existence_error(http_oauth_scope_checker,ScopeChecker)`
The scope checker `ScopeChecker` selected by `Options` does not implement `http_oauth_scope_checker_protocol`:
`domain_error(http_oauth_scope_checker,ScopeChecker)`
`Request` is not annotated with authenticated OAuth scopes:
`domain_error(http_oauth_authenticated_request,Request)`
The generated response properties `Properties` are invalid:
`domain_error(http_properties,Properties)`
The generated response header `Header` violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`
The generated response property `Property` violates normalized HTTP response semantics:
`domain_error(http_property_semantics,Property)`

---

protect_request/4

Authenticates and authorizes a normalized request.

Compilation flags:

`static`

Template:

`protect_request(Request,Verifier,Action,Options)`

Mode and number of proofs:

`protect_request(+compound,+object_identifier,--compound,+list(compound))` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`Verifier` is a variable:

`instantiation_error`

`Verifier` does not name an existing object:

`existence_error(http_oauth_verifier,Verifier)`

`Verifier` does not implement `http_oauth_verifier_protocol`:

`domain_error(http_oauth_verifier,Verifier)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The scope checker `ScopeChecker` selected by `Options` does not name an existing object:

`existence_error(http_oauth_scope_checker,ScopeChecker)`

The scope checker `ScopeChecker` selected by `Options` does not implement `http_oauth_scope_checker_protocol`:

`domain_error(http_oauth_scope_checker,ScopeChecker)`

`Options` omits a protected-resource identifier:

`domain_error(http_oauth_protection_options,Options)`

`Verifier` returns malformed normalized token information `TokenInfo`:

`domain_error(http_oauth_token_info,TokenInfo)`

The generated response properties `Properties` are invalid:

`domain_error(http_properties,Properties)`

The generated response header `Header` violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

The generated response property `Property` violates normalized HTTP response semantics:

`domain_error(http_property_semantics,Property)`

token_info/2

Returns normalized OAuth token information from an authenticated request.

Compilation flags:
`static`

Template:
`token_info(Request,TokenInfo)`
Mode and number of proofs:
`token_info(+compound,-compound)` - `zero_or_one`

---

scopes/2

Returns granted OAuth scopes from an authenticated request.

Compilation flags:
`static`

Template:
`scopes(Request,Scopes)`
Mode and number of proofs:
`scopes(+compound,-list(atom))` - `zero_or_one`

---

unauthorized_response/3

Builds a normalized 401 response and its Bearer challenge.

Compilation flags:
`static`

Template:
`unauthorized_response(Challenge,Response,Options)`
Mode and number of proofs:
`unauthorized_response(-compound,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The scope checker `ScopeChecker` selected by `Options` does not name an existing object:
`existence_error(http_oauth_scope_checker,ScopeChecker)`
The scope checker `ScopeChecker` selected by `Options` does not implement `http_oauth_scope_checker_protocol`:
`domain_error(http_oauth_scope_checker,ScopeChecker)`
The generated response properties `Properties` are invalid:
`domain_error(http_properties,Properties)`
The generated response header `Header` violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`
The generated response property `Property` violates normalized HTTP response semantics:
`domain_error(http_property_semantics,Property)`

---

unauthorized_response/4

Decorates a normalized response as a 401 Bearer response.

Compilation flags:
`static`

Template:
`unauthorized_response(Challenge,Response0,Response,Options)`
Mode and number of proofs:
`unauthorized_response(+compound,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Challenge` is not a valid normalized Bearer challenge term or contains duplicate fields:
`domain_error(http_oauth_term(challenge),Challenge)`
`Response0` is not a valid normalized HTTP response term:
`domain_error(http_response,Response0)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The scope checker `ScopeChecker` selected by `Options` does not name an existing object:
`existence_error(http_oauth_scope_checker,ScopeChecker)`
The scope checker `ScopeChecker` selected by `Options` does not implement `http_oauth_scope_checker_protocol`:
`domain_error(http_oauth_scope_checker,ScopeChecker)`

The decorated response properties `Properties` are invalid:
`domain_error(http_properties,Properties)`
The decorated response header `Header` violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`
The decorated response property `Property` violates normalized HTTP response semantics:
`domain_error(http_property_semantics,Property)`

`forbidden_response/3`

Builds a normalized 403 response and its Bearer challenge.

Compilation flags:
`static`

Template:
`forbidden_response(Challenge,Response,Options)`
Mode and number of proofs:
`forbidden_response(-compound,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The scope checker `ScopeChecker` selected by `Options` does not name an existing object:
`existence_error(http_oauth_scope_checker,ScopeChecker)`
The scope checker `ScopeChecker` selected by `Options` does not implement `http_oauth_scope_checker_protocol`:
`domain_error(http_oauth_scope_checker,ScopeChecker)`
The generated response properties `Properties` are invalid:
`domain_error(http_properties,Properties)`
The generated response header `Header` violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`
The generated response property `Property` violates normalized HTTP response semantics:
`domain_error(http_property_semantics,Property)`

---

forbidden_response/4

Decorates a normalized response as a 403 Bearer response.

Compilation flags:

`static`

Template:

`forbidden_response(Challenge,Response0,Response,Options)`

Mode and number of proofs:

`forbidden_response(+compound,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Challenge` is not a valid normalized Bearer challenge term or contains duplicate fields:

`domain_error(http_oauth_term(challenge),Challenge)`

`Response0` is not a valid normalized HTTP response term:

`domain_error(http_response,Response0)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The scope checker `ScopeChecker` selected by `Options` does not name an existing object:

`existence_error(http_oauth_scope_checker,ScopeChecker)`

The scope checker `ScopeChecker` selected by `Options` does not implement `http_oauth_scope_checker_protocol`:

`domain_error(http_oauth_scope_checker,ScopeChecker)`

The decorated response properties `Properties` are invalid:

`domain_error(http_properties,Properties)`

The decorated response header `Header` violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

The decorated response property `Property` violates normalized HTTP response semantics:

`domain_error(http_property_semantics,Property)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.132.2 `http_oauth_exact_scope_checker`

Exact, case-sensitive OAuth scope-sufficiency checker.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_oauth_scope_checker_protocol*
Uses:
list

Remarks:
(none)

Inherited public predicates:
sufficient/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.132.3 `http_oauth_helpers`

Internal shared helpers for OAuth Bearer syntax and normalized HTTP overlays.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static`

Extends:
`public` *http_authentication_helpers*
Uses:
list
user

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *valid_bearer_token/1*
  - *parse_scope_value/2*
  - *generate_scope_value/2*
  - *quoted_authentication_parameter/3*

    - *overlay_http_headers/3*
    - *overlay_http_properties/3*
  - *Private predicates*
  - *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

valid_bearer_token/1

Succeeds when the atom is a nonempty RFC 6750 b64token value.

Compilation flags:
`static`

Template:
`valid_bearer_token(Token)`
Mode and number of proofs:
`valid_bearer_token(+atom)` - `zero_or_one`

---

parse_scope_value/2

Parses and validates a space-delimited OAuth scope value.

Compilation flags:
`static`

Template:
`parse_scope_value(Value,Scopes)`
Mode and number of proofs:
`parse_scope_value(+atom,-list(atom))` - `zero_or_one`

---

generate_scope_value/2

Validates and generates a space-delimited OAuth scope value.

Compilation flags:
`static`

Template:
`generate_scope_value(Scopes,Value)`

Mode and number of proofs:
`generate_scope_value(+list(atom),-atom) - zero_or_one`

---

quoted_authentication_parameter/3

Generates a quoted HTTP authentication parameter, escaping quotes and backslashes.

Compilation flags:
`static`

Template:
`quoted_authentication_parameter(Name,Value,Parameter)`
Mode and number of proofs:
`quoted_authentication_parameter(+atom,+atom,-atom) - one`

---

overlay_http_headers/3

Overlays normalized HTTP headers by header name.

Compilation flags:
`static`

Template:
`overlay_http_headers(Overrides,Headers,OverlayHeaders)`
Mode and number of proofs:
`overlay_http_headers(+list(pair),+list(pair),-list(pair)) - one`

---

overlay_http_properties/3

Overlays normalized HTTP properties by functor and arity.

Compilation flags:
`static`

Template:
`overlay_http_properties(Overrides,Properties,OverlayProperties)`
Mode and number of proofs:
`overlay_http_properties(+list(compound),+list(compound),-list(compound)) - one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.132.4 `http_oauth_introspection_client`

OAuth token introspection HTTP client following RFC 7662.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
`public` *http_oauth_helpers*
Uses:
http_authenticate
http_core
http_oauth
list
url(Representation)

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *introspect/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

introspect/4

Introspects an access token using an authenticated form POST and returns normalized introspection data.

Compilation flags:

`static`

Template:

`introspect(Endpoint,Token,Introspection,Options)`

Mode and number of proofs:

`introspect(+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` does not include a client authentication option:

`domain_error(http_oauth_introspection_options,Options)`

`Endpoint` is not an HTTPS URL or an explicitly allowed loopback HTTP URL:

`domain_error(http_oauth_introspection_endpoint,Endpoint)`

`Token` is not a valid RFC 6750 Bearer token:

`domain_error(http_oauth_introspection_token,Token)`

The Basic client authentication username is invalid:

`domain_error(http_authenticate_value(username),ClientId)`

The introspection endpoint returned a non-success status code:

`domain_error(http_oauth_introspection_status,Code)`

The introspection response body is not JSON content:

`domain_error(http_oauth_introspection_response,Body)`

The introspection response JSON value is not an object:

`domain_error(http_oauth_introspection_response,JSON)`

The introspection response has invalid `active` member values `Values`:

`domain_error(http_oauth_introspection_active,Values)`

The introspection response has duplicate values `Values` for optional member `Name`:

`domain_error(http_oauth_introspection_member(Name),Values)`

The introspection response has invalid `scope` member value `Value`:

`domain_error(http_oauth_scope,Value)`

An option passed using `http_options/1` is not a valid HTTP client request option:

`domain_error(http_client_request_option,Option)`

The delegated HTTP client rejects the response stream:

`domain_error(http_response_stream,Error)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.132.5 `http_oauth_introspection_verifier(Endpoint,Options)`

- `Endpoint` - Token introspection endpoint URL.
- `Options` - Introspection client options plus optional `audience_validation(trust_active)`.

Resource-aware OAuth verifier using RFC 7662 token introspection.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_oauth_verifier_protocol*
Uses:
http_oauth_introspection_client
list

Remarks:
(none)

Inherited public predicates:
verify/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.132.6 `http_oauth_jwt_verifier(KeyOrJWKSet,Options)`

- `KeyOrJWKSet` - Symmetric key, public JWK, or JWK Set accepted by `jwt::verify/4`.
- `Options` - Options passed to `jwt::verify/4`.

Resource-aware OAuth verifier for JWT access tokens.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_oauth_verifier_protocol*
Imports:
`public` *http_oauth_helpers*
Uses:
jwt
list

Remarks:
(none)

Inherited public predicates:
verify/3

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.132.7 `http_oauth_metadata`

OAuth protected-resource metadata document and HTTP response builder following RFC 9728.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
http_core
list
type
url(Representation)

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *well_known_url/2*
  - *document/3*
  - *document/4*
  - *response/3*
  - *response/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

well_known_url/2

Returns the RFC 9728 well-known metadata URL for a canonical protected-resource identifier.

Compilation flags:
`static`

Template:
`well_known_url(ProtectedResource,URL)`
Mode and number of proofs:
`well_known_url(+atom,-atom)` - `one_or_error`

Exceptions:
`ProtectedResource` is not a canonical HTTPS protected-resource identifier:
`domain_error(http_oauth_protected_resource,ProtectedResource)`

---

document/3

Builds a protected-resource metadata JSON object from descriptor terms.

Compilation flags:
`static`

Template:
`document(ProtectedResource,Descriptors,Document)`
Mode and number of proofs:
`document(+atom,+list(compound),-term)` - `one_or_error`

Exceptions:
`ProtectedResource` is not a canonical HTTPS protected-resource identifier:
`domain_error(http_oauth_protected_resource,ProtectedResource)`
`Descriptors` is not a proper list:
`type_error(list,Descriptors)`
An element `Descriptor` of the list `Descriptors` is not a valid metadata descriptor:

`domain_error(http_oauth_metadata_descriptor,Descriptor)`

`Descriptors` contains duplicate metadata members:

`domain_error(http_oauth_metadata_descriptors,Descriptors)`

`document/4`

Builds a protected-resource metadata JSON object subject to profile options.

Compilation flags:

`static`

Template:

`document(ProtectedResource,Descriptors,Document,Options)`

Mode and number of proofs:

`document(+atom,+list(compound),-term,+list(compound))` - `one_or_error`

Exceptions:

`ProtectedResource` is not a canonical HTTPS protected-resource identifier:

`domain_error(http_oauth_protected_resource,ProtectedResource)`

`Descriptors` is not a proper list:

`type_error(list,Descriptors)`

An element `Descriptor` of the list `Descriptors` is not a valid metadata descriptor:

`domain_error(http_oauth_metadata_descriptor,Descriptor)`

`Descriptors` contains duplicate metadata members:

`domain_error(http_oauth_metadata_descriptors,Descriptors)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Descriptors` omits a required metadata member or gives it an empty value:

`domain_error(http_oauth_metadata_descriptors,Descriptors)`

---

response/3

Builds a normalized HTTP 200 response containing protected-resource metadata.

Compilation flags:
`static`

Template:
`response(ProtectedResource,Descriptors,Response)`
Mode and number of proofs:
`response(+atom,+list(compound),-compound)` - `one_or_error`

Exceptions:
`ProtectedResource` is not a canonical HTTPS protected-resource identifier:
`domain_error(http_oauth_protected_resource,ProtectedResource)`
`Descriptors` is not a proper list:
`type_error(list,Descriptors)`
An element `Descriptor` of the list `Descriptors` is not a valid metadata descriptor:
`domain_error(http_oauth_metadata_descriptor,Descriptor)`
`Descriptors` contains duplicate metadata members:
`domain_error(http_oauth_metadata_descriptors,Descriptors)`

---

response/4

Builds a normalized HTTP 200 metadata response subject to profile options.

Compilation flags:
`static`

Template:
`response(ProtectedResource,Descriptors,Response,Options)`
Mode and number of proofs:
`response(+atom,+list(compound),-compound,+list(compound))` - `one_or_error`

Exceptions:
`ProtectedResource` is not a canonical HTTPS protected-resource identifier:
`domain_error(http_oauth_protected_resource,ProtectedResource)`
`Descriptors` is not a proper list:
`type_error(list,Descriptors)`
An element `Descriptor` of the list `Descriptors` is not a valid metadata descriptor:
`domain_error(http_oauth_metadata_descriptor,Descriptor)`
`Descriptors` contains duplicate metadata members:
`domain_error(http_oauth_metadata_descriptors,Descriptors)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Descriptors` omits a required metadata member or gives it an empty value:
`domain_error(http_oauth_metadata_descriptors,Descriptors)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.132.8 `http_oauth_scope_checker_protocol`

Protocol for OAuth scope-sufficiency checker objects.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

  - *sufficient/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

sufficient/2

Succeeds when the granted scopes are sufficient for all required scopes.

Compilation flags:
`static`

Template:
`sufficient(GrantedScopes,RequiredScopes)`
Mode and number of proofs:
`sufficient(++list(atom),++list(atom))` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.132.9 http_oauth_verifier_protocol

Protocol for OAuth access-token verifier objects.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static`

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *verify/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

verify/3

Verifies an access token for the expected protected-resource identifier and returns normalized token information. Fails for invalid, inactive, or wrong-audience tokens.

Compilation flags:
`static`

Template:
`verify(Token,ProtectedResource,TokenInfo)`
Mode and number of proofs:
`verify(++atom,++atom,-compound)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.132.10 `http_router_oauth(Verifier,BaseOptions)`

- `Verifier` - Verifier object passed to `http_oauth::protect_request/4` for routes declaring OAuth metadata.
- `BaseOptions` - Base protection options overlaid by route-specific `oauth/1` metadata options.

Optional OAuth router helpers for objects importing the `http_router` category.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static`

Uses:
http_core
http_oauth
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *authorize_oauth_request/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

`authorize_oauth_request/2`

Applies OAuth route metadata to a routed request and returns `continue(Request)` or `respond(Response)`.

Compilation flags:
`static`

Template:
`authorize_oauth_request(Request,Action)`

Mode and number of proofs:

`authorize_oauth_request(+compound,-compound)` - `one_or_error`

Exceptions:

An OAuth route metadata or configured base options value `Options` is not a proper list:

`domain_error(http_router_oauth_options,Options)`

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

The configured verifier is a variable:

`instantiation_error`

The configured verifier `Verifier` does not name an existing object:

`existence_error(http_oauth_verifier,Verifier)`

The configured verifier `Verifier` does not implement `http_oauth_verifier_protocol`:

`domain_error(http_oauth_verifier,Verifier)`

An element of the effective OAuth protection options list is a variable:

`instantiation_error`

An element `Option` of the effective OAuth protection options list is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the effective OAuth protection options list is a compound term but not a valid OAuth option:

`domain_error(option,Option)`

The effective OAuth protection options `Options` omit a protected-resource identifier:

`domain_error(http_oauth_protection_options,Options)`

The effective OAuth scope checker is a variable:

`instantiation_error`

The effective OAuth scope checker `ScopeChecker` does not name an existing object:

`existence_error(http_oauth_scope_checker,ScopeChecker)`

The effective OAuth scope checker `ScopeChecker` does not implement `http_oauth_scope_checker_protocol`:

`domain_error(http_oauth_scope_checker,ScopeChecker)`

The OAuth verifier returns malformed normalized token information `TokenInfo`:

`domain_error(http_oauth_token_info,TokenInfo)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.132.11 http_server_core_oauth_endpoint_handler(ProtectedResource,MetadataDescriptors,MetadataOptions,Verifier,Handler,ProtectOptions)

- `ProtectedResource` - Canonical OAuth protected-resource identifier.
- `MetadataDescriptors` - Protected-resource metadata descriptor list.
- `MetadataOptions` - Options passed to `http_oauth_metadata::response/4`.
- `Verifier` - Object implementing the `http_oauth_verifier_protocol` protocol.
- `Handler` - Wrapped object implementing the `http_handler_protocol` protocol.
- `ProtectOptions` - Options passed to `http_oauth::protect_request/4`.

OAuth HTTP handler serving public protected-resource metadata and protecting all other requests.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_handler_protocol*
Uses:
http_core
http_oauth_metadata
http_server_core_oauth_handler(Verifier,Handler,ProtectOptions)
list
url(Representation)

Remarks:
(none)

Inherited public predicates:
handle/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.132.12 `http_server_core_oauth_handler(Verifier,Handler,ProtectOptions)`

- `Verifier` - Object implementing the `http_oauth_verifier_protocol` protocol.
- `Handler` - Wrapped object implementing the `http_handler_protocol` protocol.
- `ProtectOptions` - Options passed to `http_oauth::protect_request/4`.

Wrapper HTTP handler object that applies OAuth Bearer protection around another portable HTTP handler.

Availability:
`logtalk_load(http_oauth(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-01

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_handler_protocol*
Uses:
http_oauth

Remarks:
(none)

Inherited public predicates:
handle/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.133 http_parameters

object

1.133.1 `http_parameters`

Typed HTTP query, form, path, header, and cookie parameter extraction helpers plus OpenAPI descriptor generation helpers.

Availability:
`logtalk_load(http_parameters(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-09-12

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *http_json_term_helpers*
Uses:
http_core
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parameters/3*

  - *parameter/3*
  - *open_api_parameters/2*
  - *open_api_request_body/3*
  - *open_api_bad_request_response/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parameters/3

Extracts typed parameters from a normalized HTTP request using the given declaration list. Client-input failures throw `domain_error(http_parameter_validation, Errors)` where `Errors` is a non-empty list of structured parameter errors.

Compilation flags:

`static`

Template:

`parameters(Request,Declarations,Parameters)`

Mode and number of proofs:

`parameters(+compound,+list(compound),-list(compound))` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`Declarations` is a variable or a partial list:

`instantiation_error`

`Declarations` is neither a variable nor a list:

`type_error(list,Declarations)`

An element `Declaration` of the list `Declarations` is not a valid HTTP parameter declaration:

`domain_error(http_parameter_declaration,Declaration)`

An element `Declaration` of the list `Declarations` has an invalid parameter name:

`domain_error(http_parameter_name,Name)`

An element `Declaration` of the list `Declarations` has an invalid parameter source:

`domain_error(http_parameter_source,Source)`

An element `Declaration` of the list `Declarations` has an invalid parameter type:

`domain_error(http_parameter_type,Type)`

An element `Declaration` of the list `Declarations` has duplicated declaration options:

`domain_error(http_parameter_options,duplicate(Option))`

An element `Declaration` of the list `Declarations` has an invalid declaration option:

`domain_error(http_parameter_option,Option)`

An element `Declaration` of the list `Declarations` has an invalid description:

`domain_error(http_parameter_description,Description)`

An element `Declaration` of the list `Declarations` has an invalid default value:

`domain_error(http_parameter_default(Type),invalid_default(Name,Value))`

`Declarations` contains duplicated declarations for a parameter source:

`domain_error(http_parameter_declaration(Name,Source),duplicate)`

The request parameters fail declared type or constraint validation:

`domain_error(http_parameter_validation,Errors)`

---

parameter/3

Extracts a single typed parameter from a normalized HTTP request using the given declaration. Optional absent parameters fail instead of throwing an error.

Compilation flags:

`static`

Template:

`parameter(Request,Declaration,Value)`

Mode and number of proofs:

`parameter(+compound,+compound,-term)` - `zero_or_one`

---

open_api_parameters/2

Generates OpenAPI `parameter/5` descriptors for query, path, header, and cookie declarations.

Compilation flags:

`static`

Template:

`open_api_parameters(Declarations,Parameters)`

Mode and number of proofs:

`open_api_parameters(+list(compound),-list(compound))` - `one_or_error`

Exceptions:

`Declarations` is a variable or a partial list:

`instantiation_error`

`Declarations` is neither a variable nor a list:

`type_error(list,Declarations)`

An element `Declaration` of the list `Declarations` is not a valid HTTP parameter declaration:

`domain_error(http_parameter_declaration,Declaration)`

An element `Declaration` of the list `Declarations` has an invalid parameter name:

`domain_error(http_parameter_name,Name)`

An element `Declaration` of the list `Declarations` has an invalid parameter source:

`domain_error(http_parameter_source,Source)`

An element `Declaration` of the list `Declarations` has an invalid parameter type:

`domain_error(http_parameter_type,Type)`

An element `Declaration` of the list `Declarations` has an invalid declaration option:

`domain_error(http_parameter_option,Option)`

An element `Declaration` of the list `Declarations` has duplicated declaration options:

`domain_error(http_parameter_options,duplicate(Option))`
`Declarations` contains duplicated declarations for a parameter source:
`domain_error(http_parameter_declaration(Name,Source),duplicate)`

`open_api_request_body/3`

Generates an OpenAPI `request_body/3` descriptor for form declarations using the given description. Fails when the declaration list contains no form parameters.

Compilation flags:
`static`

Template:
`open_api_request_body(Declarations,Description,RequestBody)`
Mode and number of proofs:
`open_api_request_body(+list(compound),+atom,-compound)` - `zero_or_one`

`open_api_bad_request_response/1`

Returns the default OpenAPI `400 Bad Request` response descriptor used by router integrations.

Compilation flags:
`static`

Template:
`open_api_bad_request_response(Response)`
Mode and number of proofs:
`open_api_bad_request_response(-compound)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.133.2 http_router_parameters

Declarative router companion category that reuses `http_parameters` declarations for request extraction and OpenAPI metadata generation.

Availability:
`logtalk_load(http_parameters(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-08-09

Compilation flags:
`static`

Extends:
`public` *http_router*
Uses:
http_core
http_parameters
list

Remarks:

- Route declarations: Importing objects define parameter declarations per route using `route_parameter_declarations/2`.
- Handler helper: Route handlers can call `route_parameters/2` on the routed request to extract typed values using the matched route declarations.
- OpenAPI metadata: The category generates route metadata for query and path parameters, form request bodies, and a default `400 Bad Request` response descriptor from the same declarations. Importing objects can extend or override that metadata using `route_parameter_extra_metadata/2` or by overriding `route_metadata/2` and delegating with `^^route_metadata/2`.

Inherited public predicates:
api_info/1 *handle/2* *operations/1* *schema/2* *security/1* *security_scheme/2* *servers/1*

- *Public predicates*
  - *route_parameters/2*
- *Protected predicates*
  - *route_parameter_declarations/2*
  - *route_parameter_request_body_description/2*
  - *route_parameter_extra_metadata/2*
- *Private predicates*
- *Operators*

Public predicates

route_parameters/2

Extracts typed parameters for the route identified by the routed request `route/1` annotation.

Compilation flags:

`static`

Template:

`route_parameters(Request,Parameters)`

Mode and number of proofs:

`route_parameters(+compound,-list(compound))` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`Request` does not contain a route annotation:

`domain_error(http_routed_request,Request)`

The route parameter declarations are invalid:

`domain_error(http_parameter_declaration,Declaration)`

A path parameter declaration is not declared by the route path template:

```
domain_error(http_parameter_declaration(Name,path),
not_in_route_path(RouteId))
```

A path parameter declaration type is incompatible with the route path template:

```
domain_error(http_parameter_declaration(Name,path),
incompatible_route_path_type(Type,PathTemplateType))
```

The request parameters fail declared type or constraint validation:

`domain_error(http_parameter_validation,Errors)`

Protected predicates

route_parameter_declarations/2

Optional hook predicate that returns the `http_parameters` declaration list for a route identifier.

Compilation flags:

`static`

Template:

`route_parameter_declarations(RouteId,Declarations)`

Mode and number of proofs:

`route_parameter_declarations(?atom,?list(compound))` - `zero_or_one`

---

route_parameter_request_body_description/2

Optional hook predicate that customizes the generated OpenAPI request-body description for form parameter declarations.

Compilation flags:
```
static
```

Template:
```
route_parameter_request_body_description(RouteId,Description)
```
Mode and number of proofs:
```
route_parameter_request_body_description(?atom,?atom) - zero_or_one
```

---

route_parameter_extra_metadata/2

Optional hook predicate that returns additional route metadata terms to merge with the metadata generated from parameter declarations.

Compilation flags:
```
static
```

Template:
```
route_parameter_extra_metadata(RouteId,Metadata)
```
Mode and number of proofs:
```
route_parameter_extra_metadata(?atom,?list(compound)) - zero_or_one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.134 http_process_transport

object

1.134.1 http_process_transport

Process-backed HTTP transport predicates using the process library and helper processes.

Availability:
```
logtalk_load(http_process_transport(loader))
```

Author: Paulo Moura

Version: 1:5:0
Date: 2026-11-12

Compilation flags:
`static, context_switching_calls, threaded`

Implements:
`public` *http_transport_protocol*
Imports:
`public` *options*
`public` *http_message_helpers*
`public` *http_text_helpers*
Uses:
http_client_core
http_core
http_server_core
list
logtalk
os
process
reader
socket
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *close_connection/1* *close_connection_pool/1* *close_listener/1* *connection_pool_stats/2* *connection_streams/3* *default_option/1* *default_options/1* *exchange/3* *exchange/4* *exchange/5* *exchange_sequence/3* *exchange_sequence/4* *open_connection/4* *open_connection_pool/4* *open_listener/4* *option/2* *option/3* *request_listener_shutdown/1* *request_shutdown/1* *serve_listener/4* *serve_listener/5* *serve_once/3* *serve_sse_once/5* *serve_until_shutdown/4* *serve_until_shutdown/5* *serve_websocket_once/5* *supported_request_scheme/1* *supported_websocket_scheme/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *temporary_tls_credentials/3*
  - *temporary_tls_credentials_files/3*
- *Protected predicates*
- *Private predicates*
  - *process_connection_state_/3*
  - *connection_pool_seed_/1*
  - *connection_pool_config_/6*
  - *connection_pool_available_/2*
  - *connection_pool_in_use_/2*
  - *listener_seed_/1*
  - *process_listener_state_/5*
  - *listener_temporary_tls_credentials_/3*

```
      - listener_handle_alias_/2
      - listener_event_/2
      - process_listener_shutdown_requested_/1
      - listener_shutdown_seed_/1
      - listener_shutdown_control_/3
      - listener_shutdown_requested_/2
      - listener_active_worker_/3
    - Operators
```

Public predicates

temporary_tls_credentials/3

Creates or reuses temporary TLS certificate and key files under the system temporary directory using the given file name prefix.

Compilation flags:

```
static
```

Template:

```
temporary_tls_credentials(Prefix,CertificateFile,KeyFile)
```

Mode and number of proofs:

```
temporary_tls_credentials(+atom,-atom,-atom) - one_or_error
```

Exceptions:

`Prefix` is a variable:

```
instantiation_error
```

`Prefix` is not an atom:

```
type_error(atom,Prefix)
```

Temporary TLS credentials cannot be created:

```
resource_error(http_process_transport_temporary_tls_credentials)
```

---

temporary_tls_credentials_files/3

Computes the temporary TLS certificate and key file paths used for a given file name prefix without creating the files.

Compilation flags:

```
static
```

Template:

```
temporary_tls_credentials_files(Prefix,CertificateFile,KeyFile)
```

Mode and number of proofs:

```
temporary_tls_credentials_files(+atom,-atom,-atom) - one_or_error
```

Exceptions:

`Prefix` is a variable:

```
instantiation_error
```

`Prefix` is not an atom:
`type_error(atom,Prefix)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

process_connection_state_/3

Registered helper-process state for open reusable client connections.

Compilation flags:
`dynamic`

Template:
`process_connection_state_(Connection,TransportProcess,Error)`
Mode and number of proofs:
`process_connection_state_(?compound,?ground,?stream)` - `zero_or_more`

---

connection_pool_seed_/1

Last allocated connection pool identifier.

Compilation flags:
`dynamic`

Template:
`connection_pool_seed_(PoolId)`
Mode and number of proofs:
`connection_pool_seed_(?positive_integer)` - `zero_or_one`

---

connection_pool_config_/6

Stored connection pool configuration indexed by pool identifier.

Compilation flags:
`dynamic`

Template:
`connection_pool_config_(PoolId,Host,Port,MinSize,MaxSize,ConnectionOptions)`
Mode and number of proofs:

```
connection_pool_config_(?positive_integer,?atom,?integer,?non_negative_integer,?
positive_integer,?list) - zero_or_more
```

---

connection_pool_available_/2

Reusable idle client connections indexed by pool identifier.

Compilation flags:

```
dynamic
```

Template:

```
connection_pool_available_(PoolId,Connection)
```

Mode and number of proofs:

```
connection_pool_available_(?positive_integer,?compound) - zero_or_more
```

---

connection_pool_in_use_/2

Checked-out client connections indexed by pool identifier.

Compilation flags:

```
dynamic
```

Template:

```
connection_pool_in_use_(PoolId,Connection)
```

Mode and number of proofs:

```
connection_pool_in_use_(?positive_integer,?compound) - zero_or_more
```

---

listener_seed_/1

Last allocated listener identifier.

Compilation flags:

```
dynamic
```

Template:

```
listener_seed_(ListenerId)
```

Mode and number of proofs:

```
listener_seed_(?positive_integer) - zero_or_one
```

---

process_listener_state_/5

Registered process state for open listeners.

Compilation flags:
`dynamic`

Template:
`process_listener_state_(ListenerId,Executable,Process,RelayListener,Error)`
Mode and number of proofs:
`process_listener_state_(?positive_integer,?atom,?ground,?compound,?stream)` - `zero_or_more`

---

listener_temporary_tls_credentials_/3

Tracked temporary TLS credential files owned by open listeners.

Compilation flags:
`dynamic`

Template:
`listener_temporary_tls_credentials_(ListenerId,CertificateFile,KeyFile)`
Mode and number of proofs:
`listener_temporary_tls_credentials_(?positive_integer,?atom,?atom)` - `zero_or_one`

---

listener_handle_alias_/2

Maps public listener handles to registered process listener identifiers.

Compilation flags:
`dynamic`

Template:
`listener_handle_alias_(Listener,ListenerId)`
Mode and number of proofs:
`listener_handle_alias_(?term,?positive_integer)` - `zero_or_more`

---

listener_event_/2

Queued listener events parsed from the listener helper diagnostics stream.

Compilation flags:
`dynamic`

Template:
`listener_event_(ListenerId,Event)`
Mode and number of proofs:
`listener_event_(?positive_integer,?compound)` - `zero_or_more`

---

process_listener_shutdown_requested_/1

Flag recording that a synthetic listener shutdown wakeup is pending.

Compilation flags:
`dynamic`

Template:
`process_listener_shutdown_requested_(ListenerId)`
Mode and number of proofs:
`process_listener_shutdown_requested_(?positive_integer)` - `zero_or_one`

---

listener_shutdown_seed_/1

Last allocated shutdown-run identifier.

Compilation flags:
`dynamic`

Template:
`listener_shutdown_seed_(RunId)`
Mode and number of proofs:
`listener_shutdown_seed_(?positive_integer)` - `zero_or_one`

---

listener_shutdown_control_/3

Maps public shutdown control terms to listeners and active serving runs.

Compilation flags:
`dynamic`

Template:
`listener_shutdown_control_(Control,Listener,RunId)`
Mode and number of proofs:
`listener_shutdown_control_(?nonvar,?compound,?positive_integer)` - `zero_or_more`

---

listener_shutdown_requested_/2

Recorded shutdown requests for serve_until_shutdown/4 runs.

Compilation flags:
`dynamic`

Template:
`listener_shutdown_requested_(Control,RunId)`
Mode and number of proofs:
`listener_shutdown_requested_(?nonvar,?positive_integer)` - `zero_or_one`

---

listener_active_worker_/3

Tracked active connection workers for a serve_until_shutdown/4 run.

Compilation flags:
`dynamic`

Template:
`listener_active_worker_(Control,RunId,Worker)`
Mode and number of proofs:
`listener_active_worker_(?nonvar,?positive_integer,?compound)` - `zero_or_more`

Operators

(none)

1.135 http_router

category

1.135.1 `http_router`

Declarative HTTP router category for objects implementing the http_handler_protocol protocol.

Availability:
`logtalk_load(http_router(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2026-09-24

Compilation flags:
`static`

Implements:
`public` *http_handler_protocol*
Extends:
`public` *http_json_term_helpers*
`public` *http_text_helpers*
Uses:
http_core
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *handle/2*
  - *api_info/1*
  - *servers/1*
  - *security/1*
  - *operations/1*
  - *schema/2*
  - *security_scheme/2*
- *Protected predicates*
  - *middleware/2*

- response_middleware/2
- route/4
- route_metadata/2
- authorize_routed_request/2
- route_open_api_request_body_example/2
- route_open_api_response_example/2
- open_api_info/1
- open_api_servers/1
- open_api_security/1
- open_api_schema/2
- open_api_security_scheme/2
- route_produces/2
- path_template_parameters/2
- route_not_found_response/2
- route_bad_request_response/3
- route_method_not_allowed_response/3
- route_automatic_options_response/3
- route_not_acceptable_response/3

- *Private predicates*
- *Operators*

Public predicates

handle/2

Routes a normalized HTTP request using the importing object `route/4` clauses.

Compilation flags:
`static`

Template:
`handle(Request,Response)`

Mode and number of proofs:
`handle(+compound,-compound)` - `one_or_error`

Exceptions:
A middleware handler is not valid:
`domain_error(http_router_middleware,Handler)`
A middleware action is not valid:
`domain_error(http_router_middleware_action,Action)`
A response middleware handler is not valid:
`domain_error(http_router_response_middleware,Handler)`
A response middleware result is not a valid normalized HTTP response term:
`domain_error(http_router_response_middleware_response,Response)`
A route authorization action is not valid:
`domain_error(http_router_route_authorization_action,Action)`
A route `produces` declaration is not a non-empty list of media types:
`domain_error(http_router_route_produces,ProducedMediaTypes)`

A route `produces` media type is invalid:
`domain_error(http_router_route_media_type,ProducedMediaType)`
Route metadata is not a proper list:
`domain_error(http_router_route_metadata,Metadata)`
A route metadata property is invalid:
`domain_error(http_router_route_metadata_property,MetadataProperty)`
A route handler is not valid:
`domain_error(http_router_handler,Handler)`
A route path template is invalid:
`domain_error(http_router_path_template,Template)`
A route path template segment is invalid:
`domain_error(http_router_path_template_segment,Segment)`
A request path is invalid:
`domain_error(http_router_path,Path)`
The route handler reports parameter validation errors:
`domain_error(http_parameter_validation,Errors)`

`api_info/1`

Returns the top-level OpenAPI info descriptor used when a router object also implements the `open_api_provider_protocol` protocol. Importing objects can override the default descriptor by defining `open_api_info/1`.

Compilation flags:
`static`

Template:
`api_info(Info)`
Mode and number of proofs:
`api_info(-compound)` - `one`

`servers/1`

Returns the OpenAPI server descriptors used when a router object also implements the `open_api_provider_protocol` protocol. Importing objects can override the default empty list by defining `open_api_servers/1`.

Compilation flags:
`static`

Template:
`servers(Servers)`
Mode and number of proofs:
`servers(-list(compound))` - `one`

---

security/1

Optional OpenAPI top-level security hook passthrough used when a router object also implements the open_api_provider_protocol protocol. Importing objects can define open_api_security/1 to expose top-level security requirements.

Compilation flags:

static

Template:

security(Security)

Mode and number of proofs:

security(-list(compound)) - zero_or_one

---

operations/1

Derives OpenAPI operation descriptors from route/4, route_metadata/2, and route_produces/2 when a router object also implements the open_api_provider_protocol protocol. Recognized route metadata terms are summary/1, description/1, tags/1, deprecated/1, security/1, parameters/1, request_body/1, and responses/1. When request_body/1 or responses/1 metadata is absent, the router first checks optional route_open_api_request_body_example/2 and route_open_api_response_example/2 hooks and then attempts to infer inline request and response schemas by probing the route handler with a synthetic annotated request. Explicit metadata always takes precedence over inferred descriptors.

Compilation flags:

static

Template:

operations(Operations)

Mode and number of proofs:

operations(-list(compound)) - one

---

schema/2

Optional OpenAPI schema hook passthrough used when a router object also implements the open_api_provider_protocol protocol. Importing objects can define open_api_schema/2 to expose reusable schemas referenced by derived route operations.

Compilation flags:

static

Template:

schema(Name,Schema)

Mode and number of proofs:

schema(?atom,?compound) - zero_or_more

---

security_scheme/2

Optional OpenAPI security scheme hook passthrough used when a router object also implements the `open_api_provider_protocol` protocol. Importing objects can define `open_api_security_scheme/2` to expose reusable security schemes referenced by derived route operations.

Compilation flags:

```
static
```

Template:

```
security_scheme(Name,SecurityScheme)
```

Mode and number of proofs:

```
security_scheme(?atom,?compound) - zero_or_more
```

Protected predicates

middleware/2

Optional hook predicate that enumerates ordered middleware descriptors. `Id` is the middleware identifier and `Handler` is the name of a declared local predicate with arity 2, typically protected, that receives the current request and returns either `continue(Request)` or `respond(Response)`.

Compilation flags:

```
static
```

Template:

```
middleware(Id,Handler)
```

Mode and number of proofs:

```
middleware(?atom,?atom) - zero_or_more
```

---

response_middleware/2

Optional hook predicate that enumerates ordered response middleware descriptors. `Id` is the middleware identifier and `Handler` is the name of a declared local predicate with arity 3, typically protected, that receives the request used to produce the current response and returns the transformed response. Routed requests preserve their route metadata annotations and negotiated media type, when applicable.

Compilation flags:

```
static
```

Template:

```
response_middleware(Id,Handler)
```

Mode and number of proofs:

```
response_middleware(?atom,?atom) - zero_or_more
```

---

route/4

Hook predicate that enumerates route descriptors. `Id` is the route identifier, `Method` is a lowercase HTTP method atom, `PathTemplate` is a path-template atom using literal segments, anonymous `*` wildcard segments, and optional plain or typed placeholders such as `{id}`, `{id:integer}`, and `{score:number}`, and `Handler` is the name of a declared local predicate with arity 2, typically protected, that receives the annotated request and returns the response.

Compilation flags:

`static`

Template:

`route(Id,Method,PathTemplate,Handler)`

Mode and number of proofs:

`route(?atom,?atom,?atom,?atom)` - `zero_or_more`

---

route_metadata/2

Optional hook predicate that declares additional metadata properties for a route. When defined, `Metadata` must be a list of compound terms. The router removes any existing request properties with the same functors, prepends the metadata to the matched request, and then adds the standard `route/1` and `path_params/1` annotations.

Compilation flags:

`static`

Template:

`route_metadata(Id,Metadata)`

Mode and number of proofs:

`route_metadata(?atom,?list(compound))` - `zero_or_one`

---

authorize_routed_request/2

Optional hook predicate that validates or decorates a routed request after route matching and metadata annotation but before route handler dispatch. It must return either `continue(Request)` or `respond(Response)`. The default implementation continues with the routed request unchanged.

Compilation flags:

`static`

Template:

`authorize_routed_request(Request,Action)`

Mode and number of proofs:

`authorize_routed_request(+compound,-compound)` - `one_or_error`

Exceptions:

`Action` is not a valid route authorization action:

`domain_error(http_router_route_authorization_action,Action)`

---

route_open_api_request_body_example/2

Optional hook predicate that enumerates example request bodies for OpenAPI inference. Each `Body` must be either `empty` or a normalized `content(MediaType, Payload)` term. These examples are only consulted when explicit `request_body/1` route metadata is absent.

Compilation flags:

`static`

Template:

`route_open_api_request_body_example(Id,Body)`

Mode and number of proofs:

`route_open_api_request_body_example(?atom,?compound)` - `zero_or_more`

---

route_open_api_response_example/2

Optional hook predicate that enumerates example normalized HTTP responses for OpenAPI inference. These examples are only consulted when explicit `responses/1` route metadata is absent.

Compilation flags:

`static`

Template:

`route_open_api_response_example(Id,Response)`

Mode and number of proofs:

`route_open_api_response_example(?atom,?compound)` - `zero_or_more`

---

open_api_info/1

Hook predicate that importing router objects can override to customize the default top-level OpenAPI info descriptor.

Compilation flags:

`static`

Template:

`open_api_info(Info)`

Mode and number of proofs:

`open_api_info(-compound)` - `one`

---

open_api_servers/1

Hook predicate that importing router objects can override to customize the default OpenAPI server descriptors.

Compilation flags:
```
static
```

Template:
```
open_api_servers(Servers)
```
Mode and number of proofs:
```
open_api_servers(-list(compound)) - one
```

---

open_api_security/1

Optional hook predicate that importing router objects can define to expose top-level OpenAPI security requirements.

Compilation flags:
```
static
```

Template:
```
open_api_security(Security)
```
Mode and number of proofs:
```
open_api_security(-list(compound)) - zero_or_one
```

---

open_api_schema/2

Optional hook predicate that importing router objects can define to expose reusable OpenAPI schemas.

Compilation flags:
```
static
```

Template:
```
open_api_schema(Name,Schema)
```
Mode and number of proofs:
```
open_api_schema(?atom,?compound) - zero_or_more
```

---

open_api_security_scheme/2

Optional hook predicate that importing router objects can define to expose reusable OpenAPI security schemes.

Compilation flags:
`static`

Template:
`open_api_security_scheme(Name,SecurityScheme)`
Mode and number of proofs:
`open_api_security_scheme(?atom,?compound)` - `zero_or_more`

---

route_produces/2

Optional hook predicate that declares the response media types produced by a route. When defined, `MediaTypes` must be a non-empty list of media type atoms. The router negotiates the request `Accept` header against this list, annotates successful matches with `response_media_type(MediaType)`, and returns `406 Not Acceptable` when no produced type matches.

Compilation flags:
`static`

Template:
`route_produces(Id,MediaTypes)`
Mode and number of proofs:
`route_produces(?atom,?list(atom))` - `zero_or_one`

---

path_template_parameters/2

Returns the normalized placeholder descriptors declared in a path template as `parameter(Name, Type)` terms.

Compilation flags:
`static`

Template:
`path_template_parameters(PathTemplate,Parameters)`
Mode and number of proofs:
`path_template_parameters(+atom,-list(compound))` - `one_or_error`

Exceptions:
`PathTemplate` is not a valid router path template:
`domain_error(http_router_path_template,PathTemplate)`
A segment in `PathTemplate` is not a valid router path template segment:
`domain_error(http_router_path_template_segment,Segment)`

---

route_not_found_response/2

Optional hook predicate that customizes `404 Not Found` responses. When defined by the importing object, it is called with the unmatched request and must return the response to send.

Compilation flags:
`static`

Template:
`route_not_found_response(Request,Response)`
Mode and number of proofs:
`route_not_found_response(+compound,-compound)` - `zero_or_one`

---

route_bad_request_response/3

Optional hook predicate that customizes `400 Bad Request` responses generated from dedicated parameter-validation exceptions raised by route handlers. When defined by the importing object, it is called with the routed request, the structured parameter error list, and must return the response to send.

Compilation flags:
`static`

Template:
`route_bad_request_response(Request,Errors,Response)`
Mode and number of proofs:
`route_bad_request_response(+compound,+list(compound),-compound)` - `zero_or_one`

---

route_method_not_allowed_response/3

Optional hook predicate that customizes `405 Method Not Allowed` responses. When defined by the importing object, it is called with the request, the effective allowed methods list as lowercase atoms, and must return the response to send.

Compilation flags:
`static`

Template:
`route_method_not_allowed_response(Request,AllowedMethods,Response)`
Mode and number of proofs:
`route_method_not_allowed_response(+compound,+list(atom),-compound)` - `zero_or_one`

---

route_automatic_options_response/3

Optional hook predicate that customizes automatic `OPTIONS` responses when no explicit `options` route matches. When defined by the importing object, it is called with the annotated synthetic request, the effective allowed methods list as lowercase atoms, and must return the response to send.

Compilation flags:

`static`

Template:

`route_automatic_options_response(Request,EffectiveMethods,Response)`

Mode and number of proofs:

`route_automatic_options_response(+compound,+list(atom),-compound)` - `zero_or_one`

---

route_not_acceptable_response/3

Optional hook predicate that customizes `406 Not Acceptable` responses. When defined by the importing object, it is called with the request, the normalized produced media types for the matched route, and must return the response to send.

Compilation flags:

`static`

Template:

`route_not_acceptable_response(Request,ProducedMediaTypes,Response)`

Mode and number of proofs:

`route_not_acceptable_response(+compound,+list(atom),-compound)` - `zero_or_one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.136 http_server

object

1.136.1 http_server

User-facing HTTP(S) server facade built on top of the HTTP transport libraries.

Availability:
`logtalk_load(http_server(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-12

Compilation flags:
`static, context_switching_calls, threaded`

Imports:
`public` *options*
Uses:
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *open/3*
  - *open/4*
  - *close/1*
  - *request_listener_shutdown/1*
  - *serve_once/3*
  - *serve_websocket_once/5*
  - *serve/4*
  - *serve/5*
  - *serve/6*
  - *serve_until_shutdown/5*
  - *serve_until_shutdown/6*
  - *request_shutdown/1*
  - *server_property/2*
  - *start/4*
  - *start/5*
  - *stop/1*
- *Protected predicates*
- *Private predicates*
  - *server_control_/3*
  - *server_control_counter_/1*
- *Operators*

Public predicates

open/3

Opens a listener on the loopback address using the selected scheme and transport options and returns an opaque server handle.

Compilation flags:

`static`

Template:

`open(Port,Server,Options)`

Mode and number of proofs:

`open(+integer,--compound,+list)` - `one_or_error`

Exceptions:

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The selected transport object does not exist:

`existence_error(object,Transport)`

The selected transport object does not conform to the HTTP transport protocol:

`domain_error(http_transport_protocol_object,Transport)`

`Options` selects incompatible scheme and transport values:

`consistency_error(http_server_options,scheme(Scheme),transport(Transport))`

`Options` selects incompatible scheme and listener transport values:

```
consistency_error(http_server_options,scheme(Scheme),
listener_transport(ListenerTransport))
```

`Port` is not a non-negative integer or a variable accepted by the selected transport:

`domain_error(non_negative_integer,Port)`

`Options` contains listener options rejected by the selected transport:

`domain_error(option,Option)`

The selected transport cannot open the listener:

`resource_error(http_process_transport_listener)`

open/4

Opens a listener on the given host using the selected scheme and transport options and returns an opaque server handle.

Compilation flags:

`static`

Template:

`open(Host,Port,Server,Options)`

Mode and number of proofs:

`open(+atom,+integer,--compound,+list)` - `one_or_error`

Exceptions:

`Host` is not a valid listener host atom for the selected transport:

`type_error(atom,Host)`

`Port` is not a non-negative integer or a variable accepted by the selected transport:

`domain_error(non_negative_integer,Port)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The selected transport object does not exist:

`existence_error(object,Transport)`

The selected transport object does not conform to the HTTP transport protocol:

`domain_error(http_transport_protocol_object,Transport)`

`Options` selects incompatible scheme and transport values:

`consistency_error(http_server_options,scheme(Scheme),transport(Transport))`

`Options` selects incompatible scheme and listener transport values:

```
consistency_error(http_server_options,scheme(Scheme),
listener_transport(ListenerTransport))
```

`Options` contains listener options rejected by the selected transport:

`domain_error(option,Option)`

The selected transport cannot open the listener:

`resource_error(http_process_transport_listener)`

---

close/1

Closes an open server listener.

Compilation flags:
`static`

Template:
`close(Server)`
Mode and number of proofs:
`close(+compound)` - `one_or_error`

Exceptions:
`Server` is a variable:
`instantiation_error`
`Server` is not an open HTTP server handle:
`domain_error(http_server,Server)`
`Server` refers to a listener rejected by the selected transport:
`domain_error(http_socket_transport_listener,Listener)`
`Server` refers to a listener that no longer exists for the selected transport:
`existence_error(http_socket_transport_listener,Listener)`

---

request_listener_shutdown/1

Requests shutdown of a blocking accept operation on an open server listener.

Compilation flags:
`static`

Template:
`request_listener_shutdown(Server)`
Mode and number of proofs:
`request_listener_shutdown(+compound)` - `one_or_error`

Exceptions:
`Server` is a variable:
`instantiation_error`
`Server` is not an HTTP server handle:
`domain_error(http_server,Server)`
`Server` refers to a listener rejected by the selected transport:
`domain_error(http_socket_transport_listener,Listener)`
`Server` refers to a listener that no longer exists for the selected transport:
`existence_error(http_socket_transport_listener,Listener)`

---

serve_once/3

Serves a single request on an open server listener.

Compilation flags:
`static`

Template:
`serve_once(Server,Handler,ClientInfo)`
Mode and number of proofs:
`serve_once(+compound,+object_identifier,--compound)` - `one_or_error`

Exceptions:
`Server` is a variable:
`instantiation_error`
`Server` is not an open HTTP server handle:
`domain_error(http_server,Server)`
`Handler` does not conform to the HTTP handler protocol:
`domain_error(http_handler_protocol,Handler)`
The delegated HTTP server rejects the response stream:
`domain_error(http_response_stream,Error)`

---

serve_websocket_once/5

Serves a single WebSocket opening handshake on an open server listener.

Compilation flags:
`static`

Template:
`serve_websocket_once(Server,Handler,Connection,Response,ClientInfo)`
Mode and number of proofs:
`serve_websocket_once(+compound,+object_identifier,--compound,--compound,`
`--compound)` - `one_or_error`

Exceptions:
`Server` is a variable:
`instantiation_error`
`Server` is not an open HTTP server handle:
`domain_error(http_server,Server)`
`Handler` does not conform to the HTTP handler protocol:
`domain_error(http_handler_protocol,Handler)`
The WebSocket opening request does not exist:
`existence_error(http_socket_transport_websocket_request,end_of_file)`
The WebSocket opening response is invalid:
`domain_error(http_socket_transport_websocket_response,Response)`
The delegated HTTP server rejects the response stream:

`domain_error(http_response_stream,Error)`

`serve/4`

Serves a bounded number of requests on an open server, or opens a loopback listener, serves requests, and closes it.

Compilation flags:

`static`

Template:

`serve(ServerOrPort,Handler,Count,ClientInfosOrOptions)`

Mode and number of proofs:

`serve(+compound,+object_identifier,+integer,--list)` - `one_or_error`

`serve(+integer,+object_identifier,+integer,+list)` - `one_or_error`

Exceptions:

`ServerOrPort` is a variable:

`instantiation_error`

`ServerOrPort` is neither a variable nor a compound or integer:

`type_error(atom_or_compound,ServerOrPort)`

`ServerOrPort` is a compound term but not an open HTTP server handle:

`domain_error(http_server,ServerOrPort)`

`Count` is a variable:

`instantiation_error`

`Count` is not a non-negative integer:

`domain_error(non_negative_integer,Count)`

`ClientInfosOrOptions` is a variable when opening a listener from a port:

`instantiation_error`

`ClientInfosOrOptions` is neither a variable nor a list when opening a listener from a port:

`type_error(list,Options)`

An element `Option` of the list `ClientInfosOrOptions` is a variable when opening a listener from a port:

`instantiation_error`

An element `Option` of the list `ClientInfosOrOptions` is neither a variable nor a compound term when opening a listener from a port:

`type_error(compound,Option)`

An element `Option` of the list `ClientInfosOrOptions` is a compound term but not a valid option when opening a listener from a port:

`domain_error(option,Option)`

`Handler` does not conform to the HTTP handler protocol:

`domain_error(http_handler_protocol,Handler)`

The selected transport object does not exist when opening a listener from a port:

`existence_error(object,Transport)`

The selected transport object does not conform to the HTTP transport protocol when opening a listener from a port:

`domain_error(http_transport_protocol_object,Transport)`

`ClientInfosOrOptions` selects incompatible scheme and transport values when opening a listener from a port:

`consistency_error(http_server_options,scheme(Scheme),transport(Transport))`

`ClientInfosOrOptions` selects incompatible scheme and listener transport values when opening a listener from a port:

```
consistency_error(http_server_options,scheme(Scheme),
listener_transport(ListenerTransport))
```

The delegated HTTP server rejects the response stream:

```
domain_error(http_response_stream,Error)
```

Thread workers are not available on this backend:

```
resource_error(threads)
```

---

serve/5

Serves a bounded number of requests on an open server, or opens a loopback listener, serves requests, and closes it.

Compilation flags:

```
static
```

Template:

```
serve(ServerOrPort,Handler,Count,ClientInfos,Options)
```

Mode and number of proofs:

```
serve(+compound,+object_identifier,+integer,--list,+list) - one_or_error
serve(+integer,+object_identifier,+integer,--list,+list) - one_or_error
```

Exceptions:

`ServerOrPort` is a variable:

```
instantiation_error
```

`ServerOrPort` is neither a variable nor a compound or integer:

```
type_error(atom_or_compound,ServerOrPort)
```

`ServerOrPort` is a compound term but not an open HTTP server handle:

```
domain_error(http_server,ServerOrPort)
```

`Count` is a variable:

```
instantiation_error
```

`Count` is not a non-negative integer:

```
domain_error(non_negative_integer,Count)
```

`Options` is a variable:

```
instantiation_error
```

`Options` is neither a variable nor a list:

```
type_error(list,Options)
```

An element `Option` of the list `Options` is a variable:

```
instantiation_error
```

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

`Options` contains an option that is not valid for serving on an already open server:

```
domain_error(http_server_serve_option,Option)
```

`Handler` does not conform to the HTTP handler protocol:

```
domain_error(http_handler_protocol,Handler)
```

The selected transport object does not exist when opening a listener from a port:

```
existence_error(object,Transport)
```

The selected transport object does not conform to the HTTP transport protocol when opening a listener from a port:
`domain_error(http_transport_protocol_object,Transport)`
`Options` selects incompatible scheme and transport values when opening a listener from a port:
`consistency_error(http_server_options,scheme(Scheme),transport(Transport))`
`Options` selects incompatible scheme and listener transport values when opening a listener from a port:
`consistency_error(http_server_options,scheme(Scheme),listener_transport(ListenerTransport))`
`Options` contains listener-serving options rejected by the selected transport:
`domain_error(http_socket_transport_listener_option,Option)`
The delegated HTTP server rejects the response stream:
`domain_error(http_response_stream,Error)`
Thread workers are not available on this backend:
`resource_error(threads)`

---

serve/6

Opens a listener on the given host, serves a bounded number of requests, and closes it.

Compilation flags:
`static`

Template:
`serve(Host,Port,Handler,Count,ClientInfos,Options)`
Mode and number of proofs:
`serve(+atom,+integer,+object_identifier,+integer,--list,+list)` - `one_or_error`

Exceptions:
`Host` is not a valid listener host atom for the selected transport:
`type_error(atom,Host)`
`Port` is not a non-negative integer or a variable accepted by the selected transport:
`domain_error(non_negative_integer,Port)`
`Count` is a variable:
`instantiation_error`
`Count` is not a non-negative integer:
`domain_error(non_negative_integer,Count)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The selected transport object does not exist:
`existence_error(object,Transport)`

The selected transport object does not conform to the HTTP transport protocol:
`domain_error(http_transport_protocol_object,Transport)`
`Options` selects incompatible scheme and transport values:
`consistency_error(http_server_options,scheme(Scheme),transport(Transport))`
`Options` selects incompatible scheme and listener transport values:
`consistency_error(http_server_options,scheme(Scheme),listener_transport(ListenerTransport))`
`Options` contains listener-serving options rejected by the selected transport:
`domain_error(http_socket_transport_listener_option,Option)`
`Handler` does not conform to the HTTP handler protocol:
`domain_error(http_handler_protocol,Handler)`
The delegated HTTP server rejects the response stream:
`domain_error(http_response_stream,Error)`
Thread workers are not available on this backend:
`resource_error(threads)`
The selected transport cannot open or close the listener:
`resource_error(http_process_transport_listener)`

---

serve_until_shutdown/5

Opens a listener on the given host and serves requests until shutdown is requested.

Compilation flags:
`static`

Template:
`serve_until_shutdown(Host,Port,Handler,Control,Options)`
Mode and number of proofs:
`serve_until_shutdown(+atom,+integer,+object_identifier,+nonvar,+list)` - `one_or_error`

Exceptions:
`Host` is not a valid listener host atom for the selected transport:
`type_error(atom,Host)`
`Port` is not a non-negative integer or a variable accepted by the selected transport:
`domain_error(non_negative_integer,Port)`
`Control` is a variable:
`instantiation_error`
`Control` is already registered for another open-ended server loop:
`permission_error(open,http_server_shutdown_control,Control)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The selected transport object does not exist:

`existence_error(object,Transport)`

The selected transport object does not conform to the HTTP transport protocol:

`domain_error(http_transport_protocol_object,Transport)`

`Options` selects incompatible scheme and transport values:

`consistency_error(http_server_options,scheme(Scheme),transport(Transport))`

`Options` selects incompatible scheme and listener transport values:

```
consistency_error(http_server_options,scheme(Scheme),
listener_transport(ListenerTransport))
```

`Options` contains listener-serving options rejected by the selected transport:

`domain_error(http_socket_transport_listener_option,Option)`

`Handler` does not conform to the HTTP handler protocol:

`domain_error(http_handler_protocol,Handler)`

The delegated HTTP server rejects the response stream:

`domain_error(http_response_stream,Error)`

Thread workers are not available on this backend:

`resource_error(threads)`

The selected transport cannot open the listener:

`resource_error(http_process_transport_listener)`

---

serve_until_shutdown/6

Opens a listener on the given host, calls Ready after shutdown control registration, and serves requests until shutdown is requested.

Compilation flags:

`static`

Template:

`serve_until_shutdown(Host,Port,Handler,Control,Options,Ready)`

Meta-predicate template:

`serve_until_shutdown(*,*,*,*,*,0)`

Mode and number of proofs:

`serve_until_shutdown(+atom,+integer,+object_identifier,+nonvar,+list,+callable)` - `one_or_error`

Exceptions:

`Host` is not a valid listener host atom for the selected transport:

`type_error(atom,Host)`

`Port` is not a non-negative integer or a variable accepted by the selected transport:

`domain_error(non_negative_integer,Port)`

`Control` is a variable:

`instantiation_error`

`Control` is already registered for another open-ended server loop:

`permission_error(open,http_server_shutdown_control,Control)`

`Options` is a variable:
```
instantiation_error
```
`Options` is neither a variable nor a list:
```
type_error(list,Options)
```
An element `Option` of the list `Options` is a variable:
```
instantiation_error
```
An element `Option` of the list `Options` is neither a variable nor a compound term:
```
type_error(compound,Option)
```
An element `Option` of the list `Options` is a compound term but not a valid option:
```
domain_error(option,Option)
```
The selected transport object does not exist:
```
existence_error(object,Transport)
```
The selected transport object does not conform to the HTTP transport protocol:
```
domain_error(http_transport_protocol_object,Transport)
```
`Options` selects incompatible scheme and transport values:
```
consistency_error(http_server_options,scheme(Scheme),transport(Transport))
```
`Options` selects incompatible scheme and listener transport values:
```
consistency_error(http_server_options,scheme(Scheme),
listener_transport(ListenerTransport))
```
`Options` contains listener-serving options rejected by the selected transport:
```
domain_error(http_socket_transport_listener_option,Option)
```
`Handler` does not conform to the HTTP handler protocol:
```
domain_error(http_handler_protocol,Handler)
```
The delegated HTTP server rejects the response stream:
```
domain_error(http_response_stream,Error)
```
Thread workers are not available on this backend:
```
resource_error(threads)
```
The selected transport cannot open the listener:
```
resource_error(http_process_transport_listener)
```

---

request_shutdown/1

Requests shutdown of an open-ended server loop.

Compilation flags:
```
static
```

Template:
```
request_shutdown(Control)
```
Mode and number of proofs:
```
request_shutdown(+nonvar) - one_or_error
```

Exceptions:
`Control` is a variable:
```
instantiation_error
```
`Control` is not registered for an open-ended server loop:
```
existence_error(http_server_shutdown_control,Control)
```

Thread-backed shutdown is not available on this backend:
`resource_error(threads)`

---

server_property/2

Enumerates selected properties of a server handle.

Compilation flags:
`static`

Template:
`server_property(Server,Property)`
Mode and number of proofs:
`server_property(+compound,?compound)` - `zero_or_more`

---

start/4

Starts an open-ended loopback server in a worker thread and waits until it is ready to accept requests. Throws a resource error when thread support is not available.

Compilation flags:
`static`

Template:
`start(Port,Handler,Server,Options)`
Mode and number of proofs:
`start(+integer,+object_identifier,--compound,+list)` - `one_or_error`

Exceptions:
`Port` is not a non-negative integer or a variable accepted by the selected transport:
`domain_error(non_negative_integer,Port)`
`Options` specifies a variable shutdown control:
`instantiation_error`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The selected transport object does not exist:
`existence_error(object,Transport)`

The selected transport object does not conform to the HTTP transport protocol:
`domain_error(http_transport_protocol_object,Transport)`
`Options` selects incompatible scheme and transport values:
`consistency_error(http_server_options,scheme(Scheme),transport(Transport))`
`Options` selects incompatible scheme and listener transport values:
`consistency_error(http_server_options,scheme(Scheme),`
`listener_transport(ListenerTransport))`
`Options` contains listener-serving options rejected by the selected transport:
`domain_error(http_socket_transport_listener_option,Option)`
`Handler` does not conform to the HTTP handler protocol:
`domain_error(http_handler_protocol,Handler)`
The delegated HTTP server rejects the response stream:
`domain_error(http_response_stream,Error)`
The selected transport cannot open the listener:
`resource_error(http_process_transport_listener)`
Thread support is not available:
`resource_error(threads)`

`start/5`

Starts an open-ended server in a worker thread and waits until it is ready to accept requests. Throws a resource error when thread support is not available.

Compilation flags:
`static`

Template:
`start(Host,Port,Handler,Server,Options)`
Mode and number of proofs:
`start(+atom,+integer,+object_identifier,--compound,+list)` - `one_or_error`

Exceptions:
`Host` is not a valid listener host atom for the selected transport:
`type_error(atom,Host)`
`Port` is not a non-negative integer or a variable accepted by the selected transport:
`domain_error(non_negative_integer,Port)`
`Options` specifies a variable shutdown control:
`instantiation_error`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The selected transport object does not exist:

`existence_error(object,Transport)`

The selected transport object does not conform to the HTTP transport protocol:

`domain_error(http_transport_protocol_object,Transport)`

`Options` selects incompatible scheme and transport values:

`consistency_error(http_server_options,scheme(Scheme),transport(Transport))`

`Options` selects incompatible scheme and listener transport values:

```
consistency_error(http_server_options,scheme(Scheme),
listener_transport(ListenerTransport))
```

`Options` contains listener-serving options rejected by the selected transport:

`domain_error(http_socket_transport_listener_option,Option)`

`Handler` does not conform to the HTTP handler protocol:

`domain_error(http_handler_protocol,Handler)`

The delegated HTTP server rejects the response stream:

`domain_error(http_response_stream,Error)`

The selected transport cannot open the listener:

`resource_error(http_process_transport_listener)`

Thread support is not available:

`resource_error(threads)`

---

stop/1

Requests shutdown of a threaded server and waits for the worker thread to finish. Throws a resource error when thread support is not available.

Compilation flags:

`static`

Template:

`stop(Server)`

Mode and number of proofs:

`stop(+compound)` - `one_or_error`

Exceptions:

`Server` is a variable:

`instantiation_error`

`Server` is not a threaded HTTP server handle:

`domain_error(http_server,Server)`

Thread support is not available:

`resource_error(threads)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

server_control_/3

Table of HTTP server shutdown control handles.

Compilation flags:
```
dynamic
```

Template:
```
server_control_(Control,Transport,Server)
```
Mode and number of proofs:
```
server_control_(?nonvar,?object_identiifer,?compound) - zero_or_more
```

---

server_control_counter_/1

Counter for generating unique HTTP server shutdown control identifiers.

Compilation flags:
```
dynamic
```

Template:
```
server_control_counter_(Counter)
```
Mode and number of proofs:
```
server_control_counter_(-integer) - zero_or_one
```

Operators

(none)

1.137 http_server_core

object

1.137.1 http_server_core

Portable stream-based HTTP server orchestration predicates.

Availability:
```
logtalk_load(http_server_core(loader))
```

Author: Paulo Moura

Version: 1:1:1
Date: 2026-08-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *http_message_helpers*

Uses:
http_core
http_websocket_handshake
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *read_request/2*
  - *write_response/2*
  - *dispatch/3*
  - *serve/3*
  - *serve_connection/3*
  - *accept_websocket/3*
  - *serve_websocket/4*
  - *accept_sse/3*
  - *serve_sse/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

read_request/2

Reads exactly one HTTP request from a binary stream and returns it as a normalized request term. Fails on clean end-of-file before reading any bytes.

Compilation flags:
`static`

Template:
`read_request(Input,Request)`
Mode and number of proofs:
`read_request(+stream,--compound)` - `zero_or_one`

write_response/2

Writes exactly one normalized HTTP response term to a binary stream.

Compilation flags:

```
static
```

Template:

```
write_response(Output,Response)
```

Mode and number of proofs:

```
write_response(+stream,+compound) - one_or_error
```

Exceptions:

`Output` is invalid for the delegated HTTP response generator:

```
domain_error(http_sink,stream(Output))
```

`Response` is not a valid normalized HTTP response term:

```
domain_error(http_response,Response)
```

The response body file range is outside the file contents:

```
domain_error(http_body_file_range,file(File,Offset,Length))
```

---

dispatch/3

Dispatches a normalized HTTP request to a handler object implementing the http_handler_protocol protocol. Handler failures, exceptions, or invalid responses are converted to an internal server error response.

Compilation flags:

```
static
```

Template:

```
dispatch(Handler,Request,Response)
```

Mode and number of proofs:

```
dispatch(+object_identifier,+compound,-compound) - one_or_error
```

Exceptions:

---

serve/3

Reads one request from a binary input stream, dispatches it to a handler object, and writes one response to a binary output stream. HEAD requests suppress response body bytes on the wire while preserving the generated response headers. Malformed requests are converted to bad request responses.

Compilation flags:

`static`

Template:

`serve(Input,Output,Handler)`

Mode and number of proofs:

`serve(+stream,+stream,+object_identifier)` - `zero_or_one`

---

serve_connection/3

Repeatedly reads requests from a binary input stream, dispatches them to a handler object, and writes responses to a binary output stream until end-of-file, a `101 Switching Protocols` response, or connection-close semantics terminate the connection. HEAD requests suppress response body bytes on the wire while preserving the generated response headers.

Compilation flags:

`static`

Template:

`serve_connection(Input,Output,Handler)`

Mode and number of proofs:

`serve_connection(+stream,+stream,+object_identifier)` - `one_or_error`

Exceptions:

`Handler` does not conform to the HTTP handler protocol:

`domain_error(http_handler_protocol,Handler)`

`Output` is invalid for the delegated HTTP response generator:

`domain_error(http_sink,stream(Output))`

The generated response is not a valid normalized HTTP response term:

`domain_error(http_response,Response)`

The response body file range is outside the file contents:

`domain_error(http_body_file_range,file(File,Offset,Length))`

---

accept_websocket/3

Validates a normalized WebSocket opening-handshake request and builds a matching `101 Switching Protocols` response. The `protocol(Protocol)` option can be used to select a single offered subprotocol.

Compilation flags:
`static`

Template:
`accept_websocket(Request,Response,Options)`
Mode and number of proofs:
`accept_websocket(+compound,--compound,+list)` - `one_or_error`

Exceptions:
`Options` contains an invalid WebSocket acceptance option:
`domain_error(http_server_core_websocket_option,Option)`
`Options` contains reserved WebSocket handshake headers:
`domain_error(http_server_core_websocket_headers,Headers)`
`Options` contains reserved WebSocket handshake properties:
`domain_error(http_server_core_websocket_properties,Properties)`
`Request` is not a valid normalized WebSocket opening-handshake request:
`domain_error(http_server_core_websocket_request,Request)`
`Options` selects a subprotocol not offered by `Request`:
`domain_error(http_server_core_websocket_protocol,Protocol)`
The generated WebSocket response is not a valid normalized HTTP response term:
`domain_error(http_response,Response)`

Remarks:

- Repeated options: When the same WebSocket acceptance option is given multiple times, the first occurrence is used.

---

serve_websocket/4

Reads one request from a binary input stream, dispatches it to a handler object, writes one response to a binary output stream, and returns `accepted(Request, Response)` when the exchange completed with a valid WebSocket opening-handshake response. Malformed requests and non-upgrade responses are written to the stream and reported as `rejected(Response)`. Clean end-of-file before any bytes are read is reported as `end_of_file`.

Compilation flags:
`static`

Template:
`serve_websocket(Input,Output,Handler,Outcome)`
Mode and number of proofs:
`serve_websocket(+stream,+stream,+object_identifier,--compound)` - `one_or_error`

Exceptions:
`Handler` does not conform to the HTTP handler protocol:
`domain_error(http_handler_protocol,Handler)`

`Output` is invalid for the delegated HTTP response generator:
`domain_error(http_sink,stream(Output))`
The generated response is not a valid normalized HTTP response term:
`domain_error(http_response,Response)`
The response body file range is outside the file contents:
`domain_error(http_body_file_range,file(File,Offset,Length))`

---

accept_sse/3

Validates a normalized Server-Sent Events request and builds a matching streamed `200 text/event-stream` response with an empty declared body. The response headers and body are written by `serve_sse/4` without Content-Length or Transfer-Encoding framing so that the caller can continue writing an indeterminate-length event stream on the same connection.

Compilation flags:
`static`

Template:
`accept_sse(Request,Response,Options)`
Mode and number of proofs:
`accept_sse(+compound,--compound,+list)` - `one_or_error`

Exceptions:
`Options` contains an invalid SSE acceptance option:
`domain_error(http_server_core_sse_option,Option)`
`Options` contains reserved SSE response headers:
`domain_error(http_server_core_sse_headers,Headers)`
`Request` is not a valid normalized SSE request:
`domain_error(http_server_core_sse_request,Request)`
The generated SSE response is not a valid normalized HTTP response term:
`domain_error(http_response,Response)`

---

serve_sse/4

Reads one request from a binary input stream, dispatches it to a handler object, writes one response to a binary output stream, and returns `accepted(Request, Response)` when the exchange completed with a valid streamed `200 text/event-stream` response. Malformed requests and non-streaming responses are written to the stream and reported as `rejected(Response)`. Clean end-of-file before any bytes are read is reported as `end_of_file`.

Compilation flags:
`static`

Template:
`serve_sse(Input,Output,Handler,Outcome)`
Mode and number of proofs:
`serve_sse(+stream,+stream,+object_identifier,--compound)` - `one_or_error`

Exceptions:

`Handler` does not conform to the HTTP handler protocol:

```
domain_error(http_handler_protocol,Handler)
```

`Output` is invalid for the delegated HTTP response generator:

```
domain_error(http_sink,stream(Output))
```

The generated response is not a valid normalized HTTP response term:

```
domain_error(http_response,Response)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.138 http_session

object

1.138.1 `http_client_session`

By default, stateful HTTP client sessions use the `http_socket_transport` library.

Availability:

```
logtalk_load(http_session(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *http_client_session(http_socket_transport)*

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *close/1* *cookie_jar/2* *default_option/1* *default_options/1* *delete/4* *get/4* *head/4* *open/1* *open/2* *option/2* *option/3* *patch/5* *post/5* *put/5* *request/5* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.138.2 `http_client_session(HTTPTransport)`

- `HTTPTransport` - The object implementing `http_transport_protocol`.

Stateful HTTP client sessions that add cookie persistence on top of the stateless `http_client` facade.

Availability:

```
logtalk_load(http_session(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-12

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *options*
`public` *http_origin_site_helpers*

Uses:

http_client
http_cookie_jar
http_cookies(Representation)
http_core
list

Remarks:

- Option precedence: When the same session default or per-request option is given multiple times, the first occurrence is used.
- Transport options: The `connection_options(Options)` session default and request option forwards one-shot transport configuration to the underlying `http_client` facade.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *open/1*
  - *open/2*
  - *close/1*
  - *cookie_jar/2*
  - *request/5*
  - *get/4*
  - *head/4*
  - *delete/4*
  - *post/5*
  - *put/5*
  - *patch/5*
- *Protected predicates*
- *Private predicates*
  - *session_seed_/1*
  - *session_state_/2*
- *Operators*

Public predicates

open/1

Opens a new HTTP session with a freshly created owned cookie jar.

Compilation flags:

`static`

Template:

`open(Session)`

Mode and number of proofs:

`open(-compound)` - `one_or_error`

Exceptions:

open/2

Opens a new HTTP session using the given defaults and cookie-jar options, including `cookies_file(File)` for reopening a saved owned cookie jar and `connection_options(Options)` for one-shot transport configuration.

Compilation flags:
`static`

Template:
`open(Session,Options)`
Mode and number of proofs:
`open(-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Options` contains an invalid HTTP client session option:
`domain_error(http_client_session_option,Option)`
`Options` contains an invalid HTTP client session option combination:
`domain_error(http_client_session_options,Options)`
`Options` contains invalid persisted cookie data:
`domain_error(http_cookie_jar_persisted_cookies,PersistedCookies)`

---

close/1

Closes a session and, when applicable, the owned cookie jar created for it.

Compilation flags:
`static`

Template:
`close(Session)`
Mode and number of proofs:
`close(+compound)` - `one_or_error`

Exceptions:
`Session` is a variable:
`instantiation_error`
`Session` is neither a variable nor an open HTTP client session handle:
`domain_error(http_client_session,Session)`
`Session` refers to a closed HTTP client session handle:

`existence_error(http_client_session,http_client_session(SessionId))`

---

cookie_jar/2

Returns the configured cookie jar handle or the atom `none` when the session does not persist cookies.

Compilation flags:
`static`

Template:
`cookie_jar(Session,Jar)`
Mode and number of proofs:
`cookie_jar(+compound,-term)` - `one_or_error`

Exceptions:
`Session` is a variable:
`instantiation_error`
`Session` is neither a variable nor an open HTTP client session handle:
`domain_error(http_client_session,Session)`
`Session` refers to a closed HTTP client session handle:
`existence_error(http_client_session,http_client_session(SessionId))`

---

request/5

Performs one HTTP request using session defaults and automatic cookie replay/storage. Per-request options can include `connection_options(Options)` for one-shot transport configuration, `source_url(URL)` for an absolute HTTP or HTTPS URL, `source_origin(Origin)` for a bare Origin header value, and `top_level_navigation(Boolean)` to drive SameSite-aware cookie replay. When omitted, SameSite replay defaults to `source_url(URL)` and `top_level_navigation(false)`.

Compilation flags:
`static`

Template:
`request(Session,Method,URL,Response,Options)`
Mode and number of proofs:
`request(+compound,+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Session` is a variable:
`instantiation_error`
`Session` is neither a variable nor an open HTTP client session handle:
`domain_error(http_client_session,Session)`
`Session` refers to a closed HTTP client session handle:
`existence_error(http_client_session,http_client_session(SessionId))`

`Options` is a variable or a partial list:
```
instantiation_error
```
`Options` contains invalid HTTP client session request options:
```
domain_error(http_client_session_request_options,Options)
```
`Options` contains an invalid HTTP client session request option:
```
domain_error(http_client_session_request_option,Option)
```
`URL` is not a supported absolute HTTP URL:
```
domain_error(http_client_url,URL)
```
`URL` uses an unsupported HTTP scheme:
```
domain_error(http_client_scheme,Scheme)
```
Cookie replay uses an invalid request context:
```
domain_error(http_cookie_jar_request_context,RequestContext)
```
The stateless HTTP client rejects the request option:
```
domain_error(http_client_request_option,Option)
```

---

get/4

Convenience wrapper over request/5 using the `get` method.

Compilation flags:
```
static
```

Template:
```
get(Session,URL,Response,Options)
```
Mode and number of proofs:
```
get(+compound,+atom,-compound,+list(compound)) - one_or_error
```

Exceptions:
`Session` is not an open HTTP client session handle:
```
domain_error(http_client_session,Session)
```
`URL` is not a supported absolute HTTP URL:
```
domain_error(http_client_url,URL)
```
`Options` contains invalid HTTP client session request options:
```
domain_error(http_client_session_request_options,Options)
```

---

head/4

Convenience wrapper over request/5 using the `head` method.

Compilation flags:
```
static
```

Template:
```
head(Session,URL,Response,Options)
```
Mode and number of proofs:

```
head(+compound,+atom,-compound,+list(compound)) - one_or_error
```

Exceptions:

`Session` is not an open HTTP client session handle:

```
domain_error(http_client_session,Session)
```

`URL` is not a supported absolute HTTP URL:

```
domain_error(http_client_url,URL)
```

`Options` contains invalid HTTP client session request options:

```
domain_error(http_client_session_request_options,Options)
```

---

delete/4

Convenience wrapper over request/5 using the `delete` method.

Compilation flags:

```
static
```

Template:

```
delete(Session,URL,Response,Options)
```

Mode and number of proofs:

```
delete(+compound,+atom,-compound,+list(compound)) - one_or_error
```

Exceptions:

`Session` is not an open HTTP client session handle:

```
domain_error(http_client_session,Session)
```

`URL` is not a supported absolute HTTP URL:

```
domain_error(http_client_url,URL)
```

`Options` contains invalid HTTP client session request options:

```
domain_error(http_client_session_request_options,Options)
```

---

post/5

Convenience wrapper over request/5 using the `post` method and the given body.

Compilation flags:

```
static
```

Template:

```
post(Session,URL,Body,Response,Options)
```

Mode and number of proofs:

```
post(+compound,+atom,+compound,-compound,+list(compound)) - one_or_error
```

Exceptions:

`Session` is not an open HTTP client session handle:

domain_error(http_client_session,Session)
URL is not a supported absolute HTTP URL:
domain_error(http_client_url,URL)
Body is invalid for the generated normalized HTTP request:
domain_error(http_body,Body)
Options contains invalid HTTP client session request options:
domain_error(http_client_session_request_options,Options)

---

put/5

Convenience wrapper over request/5 using the put method and the given body.

Compilation flags:
static

Template:
put(Session,URL,Body,Response,Options)
Mode and number of proofs:
put(+compound,+atom,+compound,-compound,+list(compound)) - one_or_error

Exceptions:
Session is not an open HTTP client session handle:
domain_error(http_client_session,Session)
URL is not a supported absolute HTTP URL:
domain_error(http_client_url,URL)
Body is invalid for the generated normalized HTTP request:
domain_error(http_body,Body)
Options contains invalid HTTP client session request options:
domain_error(http_client_session_request_options,Options)

---

patch/5

Convenience wrapper over request/5 using the patch method and the given body.

Compilation flags:
static

Template:
patch(Session,URL,Body,Response,Options)
Mode and number of proofs:
patch(+compound,+atom,+compound,-compound,+list(compound)) - one_or_error

Exceptions:
Session is not an open HTTP client session handle:

`domain_error(http_client_session,Session)`
`URL` is not a supported absolute HTTP URL:
`domain_error(http_client_url,URL)`
`Body` is invalid for the generated normalized HTTP request:
`domain_error(http_body,Body)`
`Options` contains invalid HTTP client session request options:
`domain_error(http_client_session_request_options,Options)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

session_seed_/1

Last allocated session identifier.

Compilation flags:
`dynamic`

Template:
`session_seed_(SessionId)`
Mode and number of proofs:
`session_seed_(?positive_integer)` - `zero_or_one`

---

session_state_/2

Per-session stored cookie jar ownership and default request state.

Compilation flags:
`dynamic`

Template:
`session_state_(SessionId,State)`
Mode and number of proofs:
`session_state_(?positive_integer,?compound)` - `zero_or_more`

Operators

(none)
object

1.138.3 http_cookie_jar

HTTP cookie jar implementing explicit storage and request matching on top of the http_cookies parsing and generation predicates, with explicit save and load operations for persisting jar contents.

Availability:
`logtalk_load(http_session(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *http_text_helpers*
`public` *http_origin_site_helpers*
`public` *options*
Uses:
date
http_cookies(Representation)
list
os
url(Representation)
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *open/1*
  - *open/2*
  - *close/1*
  - *clear/1*
  - *store_set_cookies/3*
  - *request_cookies/3*
  - *request_cookies/4*
  - *cookies/2*
  - *cookie_count/2*

    - *save/2*
    - *load/2*
  - *Protected predicates*
  - *Private predicates*
    - *jar_seed_/1*
    - *jar_state_/2*
    - *jar_cookie_/11*
  - *Operators*

Public predicates

open/1

Opens a new empty cookie jar.

Compilation flags:
`static`

Template:
`open(Jar)`
Mode and number of proofs:
`open(-compound)` - `one`

---

open/2

Opens a new cookie jar using the given options list. Supported options include `cookies_file(File)` for preloading previously persisted cookies from disk.

Compilation flags:
`static`

Template:
`open(Jar,Options)`
Mode and number of proofs:
`open(-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The cookies file option is a variable:

`instantiation_error`

The cookies file option is neither a variable nor an atom:

`type_error(atom,File)`

The persisted cookie data is malformed:

`domain_error(http_cookie_jar_persisted_cookies,PersistedCookies)`

A persisted cookie is malformed:

`domain_error(http_cookie_jar_persisted_cookie,PersistedCookie)`

---

close/1

Closes a cookie jar and removes all stored cookies.

Compilation flags:

`static`

Template:

`close(Jar)`

Mode and number of proofs:

`close(+compound)` - `one_or_error`

Exceptions:

`Jar` is a variable:

`instantiation_error`

`Jar` is neither a variable nor an open cookie-jar handle:

`domain_error(http_cookie_jar,Jar)`

`Jar` refers to a closed cookie-jar handle:

`existence_error(http_cookie_jar,cookie_jar(JarId))`

---

clear/1

Removes all currently stored cookies from a cookie jar while keeping the jar handle valid.

Compilation flags:

`static`

Template:

`clear(Jar)`

Mode and number of proofs:

`clear(+compound)` - `one_or_error`

Exceptions:

`Jar` is a variable:

`instantiation_error`

`Jar` is neither a variable nor an open cookie-jar handle:

`domain_error(http_cookie_jar,Jar)`
`Jar` refers to a closed cookie-jar handle:
`existence_error(http_cookie_jar,cookie_jar(JarId))`

`store_set_cookies/3`

Stores normalized Set-Cookie terms for the given absolute URL, replacing existing cookies with the same name, domain, and path.

Compilation flags:
`static`

Template:
`store_set_cookies(Jar,URL,SetCookies)`
Mode and number of proofs:
`store_set_cookies(+compound,+atom,+list(compound))` - `one_or_error`

Exceptions:
`Jar` is a variable:
`instantiation_error`
`Jar` is neither a variable nor an open cookie-jar handle:
`domain_error(http_cookie_jar,Jar)`
`Jar` refers to a closed cookie-jar handle:
`existence_error(http_cookie_jar,cookie_jar(JarId))`
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute cookie-jar URL:
`domain_error(http_cookie_jar_url,URL)`
`URL` uses an unsupported cookie-jar URL scheme:
`domain_error(http_cookie_jar_url_scheme,Scheme)`
`SetCookies` is a variable or a partial list:
`instantiation_error`
`SetCookies` is not a valid set-cookie list:
`domain_error(http_cookie_jar_set_cookies,SetCookies)`
An element `SetCookie` of `SetCookies` is not a valid normalized Set-Cookie term:
`domain_error(http_cookie_jar_set_cookie,SetCookie)`

`request_cookies/3`

Returns the cookie name-value pairs currently applicable to the given absolute URL using the convenience default `request_context(get, source_url(URL), false)`.

Compilation flags:
`static`

Template:

```
request_cookies(Jar,URL,Cookies)
```

Mode and number of proofs:

```
request_cookies(+compound,+atom,-list(compound)) - one_or_error
```

Exceptions:

`Jar` is a variable:

```
instantiation_error
```

`Jar` is neither a variable nor an open cookie-jar handle:

```
domain_error(http_cookie_jar,Jar)
```

`Jar` refers to a closed cookie-jar handle:

```
existence_error(http_cookie_jar,cookie_jar(JarId))
```

`URL` is a variable:

```
instantiation_error
```

`URL` is not a supported absolute cookie-jar URL:

```
domain_error(http_cookie_jar_url,URL)
```

`URL` uses an unsupported cookie-jar URL scheme:

```
domain_error(http_cookie_jar_url_scheme,Scheme)
```

---

request_cookies/4

Returns the cookie name-value pairs currently applicable to the given absolute URL for an explicit request context represented as `request_context(Method, Source, TopLevelNavigation)` where `Source` is either `source_url(URL)` for an absolute HTTP or HTTPS URL or `source_origin(Origin)` for a bare Origin header value.

Compilation flags:

```
static
```

Template:

```
request_cookies(Jar,URL,RequestContext,Cookies)
```

Mode and number of proofs:

```
request_cookies(+compound,+atom,+compound,-list(compound)) - one_or_error
```

Exceptions:

`Jar` is a variable:

```
instantiation_error
```

`Jar` is neither a variable nor an open cookie-jar handle:

```
domain_error(http_cookie_jar,Jar)
```

`Jar` refers to a closed cookie-jar handle:

```
existence_error(http_cookie_jar,cookie_jar(JarId))
```

`URL` is a variable:

```
instantiation_error
```

`URL` is not a supported absolute cookie-jar URL:

```
domain_error(http_cookie_jar_url,URL)
```

`URL` uses an unsupported cookie-jar URL scheme:

```
domain_error(http_cookie_jar_url_scheme,Scheme)
```

`RequestContext` is not a valid cookie-jar request context:

```
domain_error(http_cookie_jar_request_context,RequestContext)
```

`RequestContext` contains an invalid method:
`type_error(atom,Method)`
`RequestContext` contains an invalid source:
`domain_error(http_cookie_jar_request_source,Source)`
`RequestContext` contains an invalid top-level-navigation flag:
`domain_error(boolean,TopLevelNavigation)`

---

cookies/2

Returns the currently stored cookies as cookie(Name, Value, Attributes) terms.

Compilation flags:
`static`

Template:
`cookies(Jar,Cookies)`
Mode and number of proofs:
`cookies(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Jar` is a variable:
`instantiation_error`
`Jar` is neither a variable nor an open cookie-jar handle:
`domain_error(http_cookie_jar,Jar)`
`Jar` refers to a closed cookie-jar handle:
`existence_error(http_cookie_jar,cookie_jar(JarId))`

---

cookie_count/2

Returns the number of currently stored cookies.

Compilation flags:
`static`

Template:
`cookie_count(Jar,Count)`
Mode and number of proofs:
`cookie_count(+compound,-integer)` - `one_or_error`

Exceptions:
`Jar` is a variable:
`instantiation_error`
`Jar` is neither a variable nor an open cookie-jar handle:
`domain_error(http_cookie_jar,Jar)`

`Jar` refers to a closed cookie-jar handle:
`existence_error(http_cookie_jar,cookie_jar(JarId))`

---

save/2

Persists the currently stored cookies to a file using a canonical Logtalk term representation.

Compilation flags:
`static`

Template:
`save(Jar,File)`
Mode and number of proofs:
`save(+compound,+atom)` - `one_or_error`

Exceptions:
`Jar` is a variable:
`instantiation_error`
`Jar` is neither a variable nor an open cookie-jar handle:
`domain_error(http_cookie_jar,Jar)`
`Jar` refers to a closed cookie-jar handle:
`existence_error(http_cookie_jar,cookie_jar(JarId))`
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`

---

load/2

Loads previously persisted cookies from a file, replacing the current jar contents.

Compilation flags:
`static`

Template:
`load(Jar,File)`
Mode and number of proofs:
`load(+compound,+atom)` - `one_or_error`

Exceptions:
`Jar` is a variable:
`instantiation_error`
`Jar` is neither a variable nor an open cookie-jar handle:
`domain_error(http_cookie_jar,Jar)`

`Jar` refers to a closed cookie-jar handle:
`existence_error(http_cookie_jar,cookie_jar(JarId))`
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
The persisted cookie data is malformed:
`domain_error(http_cookie_jar_persisted_cookies,PersistedCookies)`
A persisted cookie is malformed:
`domain_error(http_cookie_jar_persisted_cookie,PersistedCookie)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`jar_seed_/1`

Last allocated cookie jar identifier.

Compilation flags:
`dynamic`

Template:
`jar_seed_(JarId)`
Mode and number of proofs:
`jar_seed_(?positive_integer)` - `zero_or_one`

`jar_state_/2`

Per-jar creation index counter state.

Compilation flags:
`dynamic`

Template:
`jar_state_(JarId,CreationIndex)`
Mode and number of proofs:
`jar_state_(?positive_integer,?non_negative_integer)` - `zero_or_more`

`jar_cookie_/11`

Stored cookie entries indexed by jar identifier.

Compilation flags:
`dynamic`

Template:
`jar_cookie_(JarId,Name,Domain,Path,Value,HostOnly,Secure,HttpOnly,SameSite,Expiry,CreationIndex)`

Mode and number of proofs:
`jar_cookie_(?positive_integer,?atom,?atom,?atom,?atom,?boolean,?boolean,?boolean,?nonvar,?nonvar,?positive_integer)` - `zero_or_more`

Operators

(none)

category

1.138.4 `http_router_server_session(Manager)`

- `Manager` - Server-session manager handle used for request begin and response finish processing.

Optional server-session middleware helpers for router objects importing the `http_router` category.

Availability:
`logtalk_load(http_session(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static`

Uses:
http_server_core_session

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *annotate_server_session_request/2*

- add_server_session_response/3
- Private predicates
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

annotate_server_session_request/2

Router middleware helper that annotates a normalized request with a server-session handle and state before route dispatch.

Compilation flags:
`static`

Template:
`annotate_server_session_request(Request,Action)`
Mode and number of proofs:
`annotate_server_session_request(+compound,-compound)` - `one_or_error`

Exceptions:
The configured manager is not an open server-session manager handle:
`domain_error(http_server_core_session,Manager)`
`Request` is not a valid normalized HTTP request:
`domain_error(http_server_core_session_request,Request)`

---

add_server_session_response/3

Router response-middleware helper that finalizes server-session cookie lifecycle changes after route dispatch.

Compilation flags:
`static`

Template:
`add_server_session_response(Request,Response0,Response)`
Mode and number of proofs:
`add_server_session_response(+compound,+compound,-compound)` - `one_or_error`

Exceptions:
`Request` is not a valid annotated server-session request:
`domain_error(http_server_core_session_request,Request)`
`Response0` is not a valid normalized HTTP response:
`domain_error(http_server_core_session_response,Response0)`
The decorated response violates normalized HTTP response semantics:

```
domain_error(http_header_semantics,Header)
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.138.5 `http_server_core_session`

Server-side HTTP session manager over normalized request and response terms using opaque cookie identifiers and an in-memory session store.

Availability:
`logtalk_load(http_session(loader))`

Author: Paulo Moura
Version: 1:0:2
Date: 2026-08-18

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
date
http_cookies(Representation)
http_core
ids
list
os

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *open/1*
  - *open/2*
  - *close/1*

      - *begin/3*
      - *finish/3*
      - *current/2*
      - *ensure/2*
      - *data/2*
      - *get/3*
      - *set/3*
      - *remove/3*
      - *destroy/1*
      - *renew/2*
      - *gc/2*
      - *count/2*
  - *Protected predicates*
      - *http_server_core_session_event/2*
  - *Private predicates*
      - *manager_seed_/1*
      - *manager_state_/2*
      - *context_seed_/1*
      - *context_state_/2*
      - *stored_session_/6*
  - *Operators*

Public predicates

open/1

Opens a new server-session manager using the default cookie and timeout options.

Compilation flags:
    `static`

Template:
    `open(Manager)`
Mode and number of proofs:
    `open(-compound)` - `one_or_error`

Exceptions:

open/2

Opens a new server-session manager using the given cookie, timeout, and store options.

Compilation flags:
`static`

Template:
`open(Manager,Options)`
Mode and number of proofs:
`open(-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

close/1

Closes a server-session manager and discards all in-memory request contexts and stored sessions owned by it.

Compilation flags:
`static`

Template:
`close(Manager)`
Mode and number of proofs:
`close(+compound)` - `one_or_error`

Exceptions:
`Manager` is a variable:
`instantiation_error`
`Manager` is neither a variable nor an open server-session manager handle:
`domain_error(http_server_core_session,Manager)`

---

begin/3

Begins server-session processing for a normalized request, annotating it with a per-request session handle and the current session state.

Compilation flags:
`static`

Template:
`begin(Manager,Request,AnnotatedRequest)`
Mode and number of proofs:
`begin(+compound,+compound,-compound)` - `one_or_error`

Exceptions:
`Manager` is a variable:
`instantiation_error`
`Manager` is neither a variable nor an open server-session manager handle:
`domain_error(http_server_core_session,Manager)`
`Request` is not a valid normalized HTTP request:
`domain_error(http_server_core_session_request,Request)`

---

finish/3

Finishes server-session processing for an annotated normalized request and a normalized response, adding any needed Set-Cookie lifecycle properties.

Compilation flags:
`static`

Template:
`finish(Request,Response0,Response)`
Mode and number of proofs:
`finish(+compound,+compound,-compound)` - `one_or_error`

Exceptions:
`Request` is not a valid annotated server-session request:
`domain_error(http_server_core_session_request,Request)`
`Response0` is not a valid normalized HTTP response:
`domain_error(http_server_core_session_response,Response0)`
The decorated response violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`

---

current/2

Returns the request-bound current server-session handle from an annotated normalized request.

Compilation flags:
`static`

Template:
`current(Request,Session)`
Mode and number of proofs:
`current(+compound,-compound)` - `one_or_error`

Exceptions:
`Request` is not a valid annotated server-session request:
`domain_error(http_server_core_session_request,Request)`

---

ensure/2

Ensures that the annotated request has a backing stored server session and returns its request-bound handle.

Compilation flags:
`static`

Template:
`ensure(Request,Session)`
Mode and number of proofs:
`ensure(+compound,-compound)` - `one_or_error`

Exceptions:
`Request` is not a valid annotated server-session request:
`domain_error(http_server_core_session_request,Request)`

---

data/2

Returns the current session key-value data pairs for a request-bound server-session handle or `[]` when no backing session exists yet.

Compilation flags:
`static`

Template:
`data(Session,Data)`
Mode and number of proofs:
`data(+compound,-list(compound))` - `one_or_error`

Exceptions:

`Session` is a variable:
`instantiation_error`
`Session` is not a valid request-bound server-session handle:
`domain_error(http_server_core_session_handle,Session)`

---

get/3

Returns the current value for a session data key from a request-bound server-session handle.

Compilation flags:
`static`

Template:
`get(Session,Key,Value)`
Mode and number of proofs:
`get(+compound,+term,-term)` - `zero_or_one`

---

set/3

Sets or replaces a session data key-value pair, creating a backing stored session when needed.

Compilation flags:
`static`

Template:
`set(Session,Key,Value)`
Mode and number of proofs:
`set(+compound,+term,+term)` - `one_or_error`

Exceptions:
`Session` is a variable:
`instantiation_error`
`Session` is not a valid request-bound server-session handle:
`domain_error(http_server_core_session_handle,Session)`

---

remove/3

Removes a session data key-value pair and returns its previous value.

Compilation flags:
`static`

Template:
`remove(Session,Key,Value)`
Mode and number of proofs:
`remove(+compound,+term,-term)` - `zero_or_one`

---

destroy/1

Destroys the backing stored session, causing finish/3 to emit a deletion cookie when applicable.

Compilation flags:
`static`

Template:
`destroy(Session)`
Mode and number of proofs:
`destroy(+compound)` - `one_or_error`

Exceptions:
`Session` is a variable:
`instantiation_error`
`Session` is not a valid request-bound server-session handle:
`domain_error(http_server_core_session_handle,Session)`

---

renew/2

Renews the backing stored session identifier and returns the new opaque cookie identifier.

Compilation flags:
`static`

Template:
`renew(Session,NewIdentifier)`
Mode and number of proofs:
`renew(+compound,-atom)` - `one_or_error`

Exceptions:
`Session` is a variable:
`instantiation_error`

`Session` is not a valid request-bound server-session handle:
`domain_error(http_server_core_session_handle,Session)`

---

gc/2

Performs opportunistic garbage collection of expired stored sessions for the given manager and returns the number collected.

Compilation flags:
`static`

Template:
`gc(Manager,Collected)`
Mode and number of proofs:
`gc(+compound,-integer)` - `one_or_error`

Exceptions:
`Manager` is a variable:
`instantiation_error`
`Manager` is neither a variable nor an open server-session manager handle:
`domain_error(http_server_core_session,Manager)`

---

count/2

Returns the number of currently stored sessions owned by the given manager.

Compilation flags:
`static`

Template:
`count(Manager,Count)`
Mode and number of proofs:
`count(+compound,-integer)` - `one_or_error`

Exceptions:
`Manager` is a variable:
`instantiation_error`
`Manager` is neither a variable nor an open server-session manager handle:
`domain_error(http_server_core_session,Manager)`

Protected predicates

http_server_core_session_event/2

Optional overridable hook predicate called with server-session lifecycle events.

Compilation flags:
```
static
```

Template:
```
http_server_core_session_event(Manager,Event)
```
Mode and number of proofs:
```
http_server_core_session_event(+compound,+compound) - zero_or_one
```

Private predicates

manager_seed_/1

Last allocated server-session manager identifier.

Compilation flags:
```
dynamic
```

Template:
```
manager_seed_(ManagerId)
```
Mode and number of proofs:
```
manager_seed_(?positive_integer) - zero_or_one
```

---

manager_state_/2

Per-manager cookie, timeout, and garbage-collection state.

Compilation flags:
```
dynamic
```

Template:
```
manager_state_(ManagerId,State)
```
Mode and number of proofs:
```
manager_state_(?positive_integer,?compound) - zero_or_more
```

---

context_seed_/1

Last allocated request context identifier.

Compilation flags:
```
dynamic
```

Template:
```
context_seed_(ContextId)
```
Mode and number of proofs:
```
context_seed_(?positive_integer) - zero_or_one
```

---

context_state_/2

Per-request server-session context state.

Compilation flags:
```
dynamic
```

Template:
```
context_state_(ContextId,Context)
```
Mode and number of proofs:
```
context_state_(?positive_integer,?compound) - zero_or_more
```

---

stored_session_/6

Stored server-session data entries owned by a manager.

Compilation flags:
```
dynamic
```

Template:
```
stored_session_(ManagerId,SessionId,Data,CreatedAt,LastSeenAt,AbsoluteExpiryAt)
```
Mode and number of proofs:
```
stored_session_(?positive_integer,?atom,?list(compound),?non_negative_integer,?
non_negative_integer,?nonvar) - zero_or_more
```

Operators

(none)
object

1.138.6 `http_server_core_session_handler(Manager,Handler)`

- `Manager` - Server-session manager handle.
- `Handler` - Wrapped object implementing the `http_handler_protocol` protocol.

Wrapper HTTP handler object that applies server-session begin and finish processing around another portable HTTP handler.

Availability:
`logtalk_load(http_session(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_handler_protocol*
Uses:
http_server_core_session

Remarks:
(none)

Inherited public predicates:
handle/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.139 http_socket_transport

object

1.139.1 `http_socket_transport`

Sockets-backed HTTP transport predicates built on top of the `sockets`, `http_client_core`, and `http_server_core` libraries.

Availability:
`logtalk_load(http_socket_transport(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-08-07

Compilation flags:
`static, context_switching_calls, threaded`

Implements:
`public` *http_transport_protocol*
Imports:
`public` *options*
`public` *http_message_helpers*
`public` *http_text_helpers*
Uses:
http_client_core
http_core
http_server_core
list
socket
user

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *close_connection/1* *close_connection_pool/1* *close_listener/1* *connection_pool_stats/2* *connection_streams/3* *default_option/1* *default_options/1* *exchange/3* *exchange/4* *exchange/5* *exchange_sequence/3* *exchange_sequence/4* *open_connection/4* *open_connection_pool/4* *open_listener/4* *option/2* *option/3* *request_listener_shutdown/1* *request_shutdown/1* *serve_listener/4* *serve_listener/5* *serve_once/3* *serve_sse_once/5* *serve_until_shutdown/4* *serve_until_shutdown/5* *serve_websocket_once/5* *supported_request_scheme/1* *supported_websocket_scheme/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *connection_pool_seed_/1*
  - *connection_pool_config_/6*
  - *connection_pool_available_/2*
  - *connection_pool_in_use_/2*
  - *listener_shutdown_seed_/1*
  - *listener_shutdown_control_/3*
  - *listener_shutdown_requested_/2*
  - *listener_active_worker_/3*
  - *listener_endpoint_/3*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

connection_pool_seed_/1

Last allocated connection pool identifier.

Compilation flags:
`dynamic`

Template:
`connection_pool_seed_(PoolId)`

Mode and number of proofs:
`connection_pool_seed_(?positive_integer)` - `zero_or_one`

connection_pool_config_/6

Stored connection pool configuration indexed by pool identifier.

Compilation flags:
`dynamic`

Template:
`connection_pool_config_(PoolId,Host,Port,MinSize,MaxSize,ConnectionOptions)`
Mode and number of proofs:
`connection_pool_config_(?positive_integer,?atom,?integer,?non_negative_integer,?positive_integer,?list)` - `zero_or_more`

---

connection_pool_available_/2

Reusable idle client connections indexed by pool identifier.

Compilation flags:
`dynamic`

Template:
`connection_pool_available_(PoolId,Connection)`
Mode and number of proofs:
`connection_pool_available_(?positive_integer,?compound)` - `zero_or_more`

---

connection_pool_in_use_/2

Checked-out client connections indexed by pool identifier.

Compilation flags:
`dynamic`

Template:
`connection_pool_in_use_(PoolId,Connection)`
Mode and number of proofs:
`connection_pool_in_use_(?positive_integer,?compound)` - `zero_or_more`

---

listener_shutdown_seed_/1

Last allocated shutdown run identifier for open-ended listener loops.

Compilation flags:
`dynamic`

Template:
`listener_shutdown_seed_(RunId)`
Mode and number of proofs:
`listener_shutdown_seed_(?positive_integer)` - `zero_or_one`

---

listener_shutdown_control_/3

Registered shutdown control terms and their associated listeners.

Compilation flags:
`dynamic`

Template:
`listener_shutdown_control_(Control,Listener,RunId)`
Mode and number of proofs:
`listener_shutdown_control_(?nonvar,?compound,?positive_integer)` - `zero_or_more`

---

listener_shutdown_requested_/2

Recorded shutdown requests indexed by control term and run identifier.

Compilation flags:
`dynamic`

Template:
`listener_shutdown_requested_(Control,RunId)`
Mode and number of proofs:
`listener_shutdown_requested_(?nonvar,?positive_integer)` - `zero_or_more`

---

listener_active_worker_/3

Active listener worker records indexed by control term and run identifier.

Compilation flags:
```
dynamic
```

Template:
```
listener_active_worker_(Control,RunId,Worker)
```
Mode and number of proofs:
```
listener_active_worker_(?nonvar,?positive_integer,?compound) - zero_or_more
```

---

listener_endpoint_/3

Stored listener endpoints indexed by listener handle.

Compilation flags:
```
dynamic
```

Template:
```
listener_endpoint_(Listener,Host,Port)
```
Mode and number of proofs:
```
listener_endpoint_(?compound,?atom,?integer) - zero_or_more
```

Operators

(none)
protocol

1.139.2 http_transport_protocol

HTTP transport predicates for client connections and server listeners.

Availability:
```
logtalk_load(http_socket_transport(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-28

Compilation flags:
```
static
```

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *open_listener/4*
  - *close_listener/1*
  - *request_listener_shutdown/1*
  - *open_connection/4*
  - *close_connection/1*
  - *connection_streams/3*
  - *open_connection_pool/4*
  - *close_connection_pool/1*
  - *connection_pool_stats/2*
  - *exchange/3*
  - *exchange_sequence/3*
  - *exchange/4*
  - *exchange/5*
  - *exchange_sequence/4*
  - *serve_once/3*
  - *serve_websocket_once/5*
  - *serve_sse_once/5*
  - *serve_listener/4*
  - *serve_listener/5*
  - *serve_until_shutdown/4*
  - *serve_until_shutdown/5*
  - *request_shutdown/1*
  - *supported_request_scheme/1*
  - *supported_websocket_scheme/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

open_listener/4

Opens a TCP listener on the given host and port using the given socket options. If Port is a variable, it is unified with the bound port number.

Compilation flags:
`static`

Template:
`open_listener(Host,Port,Listener,Options)`

Mode and number of proofs:

`open_listener(+atom,?integer,--compound,+list)` - `one_or_error`

Exceptions:

`Host` is not a valid listener host atom:

`type_error(atom,Host)`

`Port` is not a non-negative integer or a variable:

`domain_error(non_negative_integer,Port)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid listener option:

`domain_error(http_process_transport_listener_option,Option)`

`Options` contains an invalid listener helper executable:

`domain_error(http_process_transport_listener_helper_executable,Executable)`

`Options` contains incompatible TCP listener options:

```
domain_error(http_process_transport_listener_tcp_options,
[listener_transport(tcp),
temporary_tls_credentials(TemporaryTLSCredentialsPrefix)])
```

`Options` contains incompatible TLS listener options:

```
domain_error(http_process_transport_listener_tls_options,
[tls_certificate_file(CertificateFile),tls_key_file(KeyFile)])
```

Temporary TLS credentials cannot be created:

`resource_error(http_process_transport_temporary_tls_credentials)`

The process-backed listener helper cannot be started:

`resource_error(http_process_transport_listener)`

---

close_listener/1

Closes a listener previously opened with open_listener/4.

Compilation flags:

`static`

Template:

`close_listener(Listener)`

Mode and number of proofs:

`close_listener(+compound)` - `one_or_error`

Exceptions:

`Listener` is a variable:

`instantiation_error`

`Listener` is not an open listener handle for the transport implementation:
`domain_error(http_socket_transport_listener,Listener)`
`Listener` no longer exists for the transport implementation:
`existence_error(http_socket_transport_listener,Listener)`

---

request_listener_shutdown/1

Best-effort wakeup request for a listener accept loop opened with `open_listener/4`. This predicate does not close the listener but causes a blocked accept call to return portably when possible.

Compilation flags:
`static`

Template:
`request_listener_shutdown(Listener)`
Mode and number of proofs:
`request_listener_shutdown(+compound)` - `one_or_error`

Exceptions:
`Listener` is a variable:
`instantiation_error`
`Listener` is not an open listener handle for the transport implementation:
`domain_error(http_socket_transport_listener,Listener)`
`Listener` no longer exists for the transport implementation:
`existence_error(http_socket_transport_listener,Listener)`

---

open_connection/4

Opens a reusable client connection to the given host and port using the given socket options.

Compilation flags:
`static`

Template:
`open_connection(Host,Port,Connection,Options)`
Mode and number of proofs:
`open_connection(+atom,+integer,--compound,+list)` - `one_or_error`

Exceptions:
`Host` and `Port` are not a valid HTTP authority target:
`domain_error(http_target,authority(Host,Port))`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Options` contains an invalid connection option:
`domain_error(http_process_transport_connection_option,Option)`
The process-backed client helper cannot be started:
`resource_error(http_process_transport_connection)`

---

close_connection/1

Closes a reusable client connection previously opened with open_connection/4.

Compilation flags:
`static`

Template:
`close_connection(Connection)`
Mode and number of proofs:
`close_connection(+compound)` - `one_or_error`

Exceptions:
`Connection` is a variable:
`instantiation_error`
`Connection` is not an open reusable client connection handle:
`domain_error(http_socket_transport_connection,Connection)`

---

connection_streams/3

Returns the binary input and output streams carried by a reusable client connection handle or by an upgraded WebSocket connection handle returned by `serve_websocket_once/5`.

Compilation flags:
`static`

Template:
`connection_streams(Connection,Input,Output)`
Mode and number of proofs:
`connection_streams(+compound,--stream,--stream)` - `one_or_error`

Exceptions:
`Connection` is a variable:
`instantiation_error`
`Connection` is not an open reusable client or upgraded WebSocket connection handle:
`domain_error(http_socket_transport_connection,Connection)`

open_connection_pool/4

Opens a managed reusable connection pool for the given host and port.

Compilation flags:

`static`

Template:

`open_connection_pool(Host,Port,Pool,Options)`

Mode and number of proofs:

`open_connection_pool(+atom,+integer,--compound,+list)` - `one_or_error`

Exceptions:

`Host` and `Port` are not a valid HTTP authority target:

`domain_error(http_target,authority(Host,Port))`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid connection-pool option:

`domain_error(http_socket_transport_connection_pool_option,Option)`

`Options` contains an invalid process-backed connection-pool option:

`domain_error(http_process_transport_connection_pool_option,Option)`

`Options` contains incompatible connection-pool bounds:

```
domain_error(http_socket_transport_connection_pool_options,
[min_size(MinSize),max_size(MaxSize)])
```

`Options` contains incompatible process-backed connection-pool bounds:

```
domain_error(http_process_transport_connection_pool_options,
[min_size(MinSize),max_size(MaxSize)])
```

No managed connection is available in the connection pool:

`resource_error(http_socket_transport_connection_pool)`

The process-backed client helper cannot be started:

`resource_error(http_process_transport_connection)`

Remarks:

- Option `min_size(N)`: Pre-open N reusable client connections when creating the pool. The default is `0`.
- Option `max_size(N)`: Allow at most N managed client connections in the pool. The default is `10`.
- Option `connection_options(Options)`: Socket options passed to `open_connection/4` when creating pooled connections. The default is `[]`.

close_connection_pool/1

Closes a managed reusable connection pool and all currently available pooled connections. Throws if pooled exchanges are still in progress.

Compilation flags:
`static`

Template:
`close_connection_pool(Pool)`
Mode and number of proofs:
`close_connection_pool(+compound)` - `one_or_error`

Exceptions:
`Pool` is a variable:
`instantiation_error`
`Pool` is not a valid managed connection-pool handle:
`domain_error(http_socket_transport_connection_pool,Pool)`
`Pool` no longer exists:
`existence_error(http_socket_transport_connection_pool,Pool)`
`Pool` cannot be closed while pooled exchanges remain active:
`permission_error(close,http_socket_transport_connection_pool,Pool)`

---

connection_pool_stats/2

Returns managed pool statistics as stats(Available, InUse, Total, MinSize, MaxSize).

Compilation flags:
`static`

Template:
`connection_pool_stats(Pool,Stats)`
Mode and number of proofs:
`connection_pool_stats(+compound,--compound)` - `one_or_error`

Exceptions:
`Pool` is a variable:
`instantiation_error`
`Pool` is not a valid managed connection-pool handle:
`domain_error(http_socket_transport_connection_pool,Pool)`
`Pool` no longer exists:
`existence_error(http_socket_transport_connection_pool,Pool)`

---

exchange/3

Performs a single HTTP exchange on an open reusable client connection or by temporarily acquiring a pooled reusable client connection.

Compilation flags:

```
static
```

Template:

```
exchange(ConnectionOrPool,Request,Response)
```

Mode and number of proofs:

```
exchange(+compound,+compound,--compound) - one_or_error
```

Exceptions:

`ConnectionOrPool` is a variable:

```
instantiation_error
```

`ConnectionOrPool` is neither an open connection nor a managed connection-pool handle:

```
domain_error(http_socket_transport_connection,ConnectionOrPool)
```

`ConnectionOrPool` is not a valid managed connection-pool handle:

```
domain_error(http_socket_transport_connection_pool,ConnectionOrPool)
```

`ConnectionOrPool` no longer exists as a managed connection pool:

```
existence_error(http_socket_transport_connection_pool,ConnectionOrPool)
```

No managed connection is available in the connection pool:

```
resource_error(http_socket_transport_connection_pool)
```

`Request` is not a valid normalized HTTP request term:

```
domain_error(http_request,Request)
```

The response stream cannot be parsed as a normalized HTTP response:

```
domain_error(http_response_stream,Error)
```

---

exchange_sequence/3

Performs a sequence of HTTP exchanges on an open reusable client connection or by temporarily acquiring a pooled reusable client connection while HTTP persistence rules allow it.

Compilation flags:

```
static
```

Template:

```
exchange_sequence(ConnectionOrPool,Requests,Responses)
```

Mode and number of proofs:

```
exchange_sequence(+compound,++list(compound),--list(compound)) - one_or_error
```

Exceptions:

`ConnectionOrPool` is a variable:

```
instantiation_error
```

`ConnectionOrPool` is neither an open connection nor a managed connection-pool handle:

```
domain_error(http_socket_transport_connection,ConnectionOrPool)
```

`ConnectionOrPool` is not a valid managed connection-pool handle:
`domain_error(http_socket_transport_connection_pool,ConnectionOrPool)`
`ConnectionOrPool` no longer exists as a managed connection pool:
`existence_error(http_socket_transport_connection_pool,ConnectionOrPool)`
No managed connection is available in the connection pool:
`resource_error(http_socket_transport_connection_pool)`
`Requests` is not a list of normalized HTTP request terms:
`domain_error(http_request_sequence,Requests)`
The response stream cannot be parsed as a normalized HTTP response:
`domain_error(http_response_stream,Error)`

---

exchange/4

Opens a client socket connection to the given host and port, performs a single HTTP exchange, and closes the connection. When the request does not already specify connection handling, `Connection: close` is added automatically.

Compilation flags:
`static`

Template:
`exchange(Host,Port,Request,Response)`
Mode and number of proofs:
`exchange(+atom,+integer,+compound,--compound)` - `one_or_error`

Exceptions:
`Host` and `Port` are not a valid HTTP authority target:
`domain_error(http_target,authority(Host,Port))`
The process-backed client helper cannot be started:
`resource_error(http_process_transport_connection)`
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
The response stream cannot be parsed as a normalized HTTP response:
`domain_error(http_response_stream,Error)`

---

exchange/5

Opens a client socket connection to the given host and port using the given connection options, performs a single HTTP exchange, and closes the connection. When the request does not already specify connection handling, `Connection: close` is added automatically.

Compilation flags:
`static`

Template:
`exchange(Host,Port,Request,Response,Options)`

Mode and number of proofs:

`exchange(+atom,+integer,+compound,--compound,+list)` - `one_or_error`

Exceptions:

`Host` and `Port` are not a valid HTTP authority target:

`domain_error(http_target,authority(Host,Port))`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid connection option:

`domain_error(http_process_transport_connection_option,Option)`

The process-backed client helper cannot be started:

`resource_error(http_process_transport_connection)`

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

The response stream cannot be parsed as a normalized HTTP response:

`domain_error(http_response_stream,Error)`

---

exchange_sequence/4

Opens a client socket connection to the given host and port, performs a sequence of HTTP exchanges on that connection, and closes the connection. When the last request does not already specify connection handling, `Connection: close` is added automatically to that final request.

Compilation flags:

`static`

Template:

`exchange_sequence(Host,Port,Requests,Responses)`

Mode and number of proofs:

`exchange_sequence(+atom,+integer,++list(compound),--list(compound))` - `one_or_error`

Exceptions:

`Host` and `Port` are not a valid HTTP authority target:

`domain_error(http_target,authority(Host,Port))`

The process-backed client helper cannot be started:

`resource_error(http_process_transport_connection)`

`Requests` is not a list of normalized HTTP request terms:

`domain_error(http_request_sequence,Requests)`

The response stream cannot be parsed as a normalized HTTP response:

`domain_error(http_response_stream,Error)`

---

serve_once/3

Accepts one incoming socket connection on the given server socket, serves that connection using the http_server_core library, closes the streams, and returns client information.

Compilation flags:

`static`

Template:

`serve_once(ServerSocket,Handler,ClientInfo)`

Mode and number of proofs:

`serve_once(+compound,+object_identifier,--compound)` - `one_or_error`

Exceptions:

`Handler` does not conform to the HTTP handler protocol:

`domain_error(http_handler_protocol,Handler)`

The delegated HTTP server rejects the response stream:

`domain_error(http_response_stream,Error)`

---

serve_websocket_once/5

Accepts one incoming socket connection on the given listener, serves exactly one WebSocket opening handshake using the given handler, and on success returns an upgraded connection handle that remains open together with the handshake response and client information. Rejected or malformed handshakes are written to the stream, the accepted streams are closed, and the predicate throws.

Compilation flags:

`static`

Template:

`serve_websocket_once(Listener,Handler,Connection,Response,ClientInfo)`

Mode and number of proofs:

`serve_websocket_once(+compound,+object_identifier,--compound,--compound,`
`--compound)` - `one_or_error`

Exceptions:

`Handler` does not conform to the HTTP handler protocol:

`domain_error(http_handler_protocol,Handler)`

The WebSocket opening request does not exist:

`existence_error(http_socket_transport_websocket_request,end_of_file)`

The WebSocket opening response is invalid:

`domain_error(http_socket_transport_websocket_response,Response)`

The delegated HTTP server rejects the response stream:

`domain_error(http_response_stream,Error)`

---

serve_sse_once/5

Accepts one incoming socket connection on the given listener, serves exactly one Server-Sent Events request using the given handler, and on success returns a streamed connection handle that remains open together with the accepted response and client information. Rejected or malformed requests are written to the stream, the accepted streams are closed, and the predicate throws.

Compilation flags:

`static`

Template:

`serve_sse_once(Listener,Handler,Connection,Response,ClientInfo)`

Mode and number of proofs:

`serve_sse_once(+compound,+object_identifier,--compound,--compound,--compound)` - `one_or_error`

Exceptions:

`Handler` does not conform to the HTTP handler protocol:

`domain_error(http_handler_protocol,Handler)`

The SSE opening request does not exist:

`existence_error(http_socket_transport_sse_request,end_of_file)`

The SSE opening response is invalid:

`domain_error(http_socket_transport_sse_response,Response)`

The delegated HTTP server rejects the response stream:

`domain_error(http_response_stream,Error)`

---

serve_listener/4

Accepts and serves Count incoming connections on the given listener, returning the accepted client information terms in order. A `request_listener_shutdown/1` call for the listener may interrupt the bounded loop before Count connections are accepted.

Compilation flags:

`static`

Template:

`serve_listener(Listener,Handler,Count,ClientInfos)`

Mode and number of proofs:

`serve_listener(+compound,+object_identifier,+integer,--list(compound))` - `one_or_error`

Exceptions:

`Count` is a variable:

`instantiation_error`

`Count` is not a non-negative integer:

`domain_error(non_negative_integer,Count)`

`Handler` does not conform to the HTTP handler protocol:

`domain_error(http_handler_protocol,Handler)`

The delegated HTTP server rejects the response stream:

`domain_error(http_response_stream,Error)`

Thread workers are not available on this backend:
`resource_error(threads)`

---

serve_listener/5

Accepts and serves `Count` incoming connections on the given listener using the specified shutdown and worker options. A `request_listener_shutdown/1` call for the listener may interrupt the bounded loop before `Count` connections are accepted.

Compilation flags:
`static`

Template:
`serve_listener(Listener,Handler,Count,ClientInfos,Options)`

Mode and number of proofs:
`serve_listener(+compound,+object_identifier,+integer,--list(compound),+list)` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is not a non-negative integer:
`domain_error(non_negative_integer,Count)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Options` contains an invalid listener-serving option:
`domain_error(http_socket_transport_listener_option,Option)`
`Handler` does not conform to the HTTP handler protocol:
`domain_error(http_handler_protocol,Handler)`
The delegated HTTP server rejects the response stream:
`domain_error(http_response_stream,Error)`
Thread workers are not available on this backend:
`resource_error(threads)`

Remarks:

- Option `shutdown(keep_open)`: Leave the listener open after serving the requested number of connections. This is the default.
- Option `shutdown(close)`: Close the listener when serving completes or aborts.
- Option `workers(serial)`: Serve accepted connections sequentially in the caller thread. This is the default.
- Option `workers(per_connection)`: Spawn one worker thread per accepted connection and wait for all workers before returning. Requires backend thread support.

- Option `workers(pool(Size))`: Serve accepted connections in batches of up to `Size` worker threads, waiting for each batch to finish before accepting the next batch. Requires backend thread support.
- Option `workers(pool(Size, rolling))`: Serve accepted connections using at most `Size` concurrent worker threads, accepting the next connection as soon as a worker finishes. Requires backend thread support.

---

serve_until_shutdown/4

Accepts and serves incoming connections on the given listener until `request_shutdown/1` is called for the specified control term, then closes the listener before returning.

Compilation flags:

`static`

Template:

`serve_until_shutdown(Listener,Handler,Control,Options)`

Mode and number of proofs:

`serve_until_shutdown(+compound,+object_identifier,+nonvar,+list)` - `one_or_error`

Exceptions:

`Control` is a variable:

`instantiation_error`

`Control` is already registered for another open-ended listener loop:

`permission_error(reuse,http_socket_transport_shutdown_control,Control)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

`Options` contains an invalid listener-serving option:

`domain_error(http_socket_transport_listener_option,Option)`

`Handler` does not conform to the HTTP handler protocol:

`domain_error(http_handler_protocol,Handler)`

The delegated HTTP server rejects the response stream:

`domain_error(http_response_stream,Error)`

Thread workers are not available on this backend:

`resource_error(threads)`

Remarks:

- Control term: The control term must be non-variable and should be fresh for each open-ended serving loop.
- Option `workers(serial)`: Serve accepted connections sequentially in the caller thread. This is the default.
- Option `workers(per_connection)`: Spawn one worker thread per accepted connection and wait for all workers when shutdown is requested. Requires backend thread support.
- Option `workers(pool(Size))`: Serve accepted connections using at most Size concurrent worker threads, waiting for worker completion notifications before accepting additional connections. Requires backend thread support.
- Option `workers(pool(Size, rolling))`: Alias for `workers(pool(Size))`. Requires backend thread support.
- Shutdown behavior: Calling request_shutdown/1 wakes the serving loop, stops accepting new connections, and waits for active workers to finish before the listener is closed and the serving loop returns.

---

serve_until_shutdown/5

Accepts and serves incoming connections on the given listener until `request_shutdown/1` is called for the specified control term, calling Ready after the shutdown control is registered and before the serving loop starts accepting connections.

Compilation flags:

`static`

Template:

`serve_until_shutdown(Listener,Handler,Control,Options,Ready)`

Meta-predicate template:

`serve_until_shutdown(*,*,*,*,0)`

Mode and number of proofs:

`serve_until_shutdown(+compound,+object_identifier,+nonvar,+list,+callable)` - `one_or_error`

Exceptions:

`Control` is a variable:

`instantiation_error`

`Control` is already registered for another open-ended listener loop:

`permission_error(reuse,http_socket_transport_shutdown_control,Control)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

`Options` contains an invalid listener-serving option:

`domain_error(http_socket_transport_listener_option,Option)`

`Handler` does not conform to the HTTP handler protocol:

`domain_error(http_handler_protocol,Handler)`

The delegated HTTP server rejects the response stream:

`domain_error(http_response_stream,Error)`

Thread workers are not available on this backend:

`resource_error(threads)`

Remarks:

- Control term: The control term must be non-variable and should be fresh for each open-ended serving loop.
- Ready goal: Called once after the shutdown control is registered and before the serving loop starts accepting connections.
- Options: Supports the same worker options as `serve_until_shutdown/4`.

request_shutdown/1

Requests shutdown of an open-ended serving loop started with `serve_until_shutdown/4` or `serve_until_shutdown/5` for the specified control term and wakes any blocked accept call so the loop can terminate portably.

Compilation flags:

`static`

Template:

`request_shutdown(Control)`

Mode and number of proofs:

`request_shutdown(+nonvar)` - `one_or_error`

Exceptions:

`Control` is a variable:

`instantiation_error`

`Control` is not registered for an open-ended listener loop:

`existence_error(http_socket_transport_shutdown_control,Control)`

Thread-backed shutdown is not available on this backend:

`resource_error(threads)`

---

supported_request_scheme/1

Table of supported request schemas.

Compilation flags:

`static`

Template:

`supported_request_scheme(Scheme)`

Mode and number of proofs:

`supported_request_scheme(+nonvar)` - `one_or_more`

---

supported_websocket_scheme/1

Table of supported websocket schemas.

Compilation flags:

`static`

Template:

`supported_websocket_scheme(Scheme)`

Mode and number of proofs:

`supported_websocket_scheme(+nonvar)` - `one_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.140 http_sse

object

1.140.1 `http_sse`

High-level Server-Sent Events (SSE) predicates for opening client connections, accepting server connections, exchanging events, and running common client and server session loops.

Availability:
`logtalk_load(http_sse(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
atom
http_client
json
list
term_io
url(Representation)
user
utf_8

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *open/2*
  - *open/3*
  - *accept/3*
  - *accept/4*
  - *send/2*
  - *send/3*
  - *receive/2*
  - *receive/3*
  - *close/1*
  - *close/2*
  - *property/2*
  - *send_data/2*
  - *send_event/3*
  - *send_id_data/3*
  - *send_comment/2*
  - *send_retry/2*
  - *send_json/2*
  - *receive_json/2*
  - *send_term/2*
  - *receive_term/2*
  - *receive_data/2*
  - *open_session/4*
  - *open_session/5*
  - *serve_once/5*
  - *serve_once/6*
- *Protected predicates*
- *Private predicates*
  - *handle_seed_/1*
  - *handle_state_/9*
- *Operators*

Public predicates

open/2

Opens a client SSE connection to the given absolute `http://` or `https://` URL and returns an opaque handle managed by this object. Equivalent to `open/3` with an empty options list.

Compilation flags:

`static`

Template:

`open(URL,SSE)`

Mode and number of proofs:

`open(+atom,--compound)` - `one_or_error`

Exceptions:

`URL` is a variable:
`instantiation_error`
`URL` is neither a variable nor a supported absolute SSE URL:
`domain_error(http_client_sse_url,URL)`
`URL` is a supported absolute SSE URL but uses an unsupported SSE scheme:
`domain_error(http_client_sse_scheme,Scheme)`
The SSE server response is not a `200 text/event-stream` response:
`domain_error(http_client_sse_response,Response)`

---

open/3

Opens a client SSE connection to the given URL and returns an opaque handle. The `transport/1` option selects the transport, with `transport(default)` deriving it from the URL scheme (`http` uses `http_socket_transport` and `https` uses `http_process_transport`). The `max_field_length/1` option bounds the length, in bytes, of any single field line read from the connection (default `none`, meaning unbounded). Remaining options, including `headers/1`, `query/1`, `version/1`, `connection_options/1`, and `last_event_id/1` (used to populate an initial `Last-Event-ID` request header and the handle `last_event_id/1` property), are forwarded to `http_client::open_sse/4`.

Compilation flags:
`static`

Template:
`open(URL,SSE,Options)`
Mode and number of proofs:
`open(+atom,--compound,+list)` - `one_or_error`

Exceptions:
`URL` is a variable:
`instantiation_error`
`URL` is neither a variable nor a supported absolute SSE URL:
`domain_error(http_client_sse_url,URL)`
`URL` is a supported absolute SSE URL but uses an unsupported SSE scheme:
`domain_error(http_client_sse_scheme,Scheme)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
`Options` contains an invalid direct SSE option:
`domain_error(http_sse_option,Option)`
`Options` contains an invalid SSE client option:
`domain_error(http_client_sse_option,Option)`
The SSE server response is not a `200 text/event-stream` response:
`domain_error(http_client_sse_response,Response)`

---

accept/3

Accepts one incoming SSE request on the given listener using the default response policy and returns an opaque server-side handle together with the accepted client information. Equivalent to `accept/4` with an empty options list.

Compilation flags:

`static`

Template:

`accept(Listener,SSE,ClientInfo)`

Mode and number of proofs:

`accept(+compound,--compound,--compound)` - `one_or_error`

Exceptions:

The SSE request does not exist:

`existence_error(http_socket_transport_sse_request,end_of_file)`

The SSE request is not a valid normalized SSE request:

`domain_error(http_server_core_sse_request,Request)`

---

accept/4

Accepts one incoming SSE request on the given listener and returns an opaque server-side handle. The `transport/1` and `max_field_length/1` options are interpreted as in `open/3`. The `retry/1` option (default `none`), when not `none`, writes a `retry(Milliseconds)` record, as accepted by `send/2`, immediately after acceptance, before returning; this is equivalent to, but saves, a first call to `send_retry/2`. Remaining options, including `headers/1` (extra response headers) and `properties/1` (extra response properties), are forwarded to `http_server_core::accept_sse/3`.

Compilation flags:

`static`

Template:

`accept(Listener,SSE,ClientInfo,Options)`

Mode and number of proofs:

`accept(+compound,--compound,--compound,+list)` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

`Options` contains an invalid direct SSE option:

`domain_error(http_sse_option,Option)`

`Options` contains an invalid SSE acceptance option:

`domain_error(http_server_core_sse_option,Option)`

`Options` contains reserved SSE response headers:

`domain_error(http_server_core_sse_headers,Headers)`

The SSE request does not exist:

`existence_error(http_socket_transport_sse_request,end_of_file)`

The SSE request is not a valid normalized SSE request:
`domain_error(http_server_core_sse_request,Request)`

---

send/2

Writes one outbound SSE record using the opaque handle. Accepts normalized `event(Type, Data, Id)` terms (`Type` and `Id` are the atom `none` when absent) and the convenience wrappers `data(Data)`, `event(Type, Data)`, `id_data(Id, Data)`, `comment(Text)`, `retry(Milliseconds)`, `json(JSON)`, and `term(Term)`. Equivalent to `send/3` with an empty options list.

Compilation flags:
`static`

Template:
`send(SSE,Event)`
Mode and number of proofs:
`send(+compound,+term)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`SSE` is an open opaque SSE handle but refers to a closed opaque SSE handle:
`existence_error(http_sse_handle,SSE)`
`Event` is not a valid outbound SSE record:
`domain_error(http_sse_event,Event)`

---

send/3

Writes one outbound SSE record using the opaque handle and the given write options. The direct API accepts a `flush(on|off)` option controlling whether the underlying stream is flushed after writing (default `on`).

Compilation flags:
`static`

Template:
`send(SSE,Event,Options)`
Mode and number of proofs:
`send(+compound,+term,+list)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:

`domain_error(http_sse_handle,SSE)`

`SSE` is an open opaque SSE handle but refers to a closed opaque SSE handle:

`existence_error(http_sse_handle,SSE)`

`Event` is not a valid outbound SSE record:

`domain_error(http_sse_event,Event)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

`Options` contains an invalid write option:

`domain_error(http_sse_write_option,Option)`

`receive/2`

Reads and dispatches the next SSE event using the opaque handle, following the event stream interpretation algorithm. Comment lines and `retry` fields are consumed silently and, for `retry`, update the handle `retry/1` property. Returns `end_of_file`, and closes and deregisters the handle, when the peer closes the connection before another event is dispatched. Equivalent to `receive/3` with an empty options list.

Compilation flags:

`static`

Template:

`receive(SSE,Event)`

Mode and number of proofs:

`receive(+compound,--term)` - `one_or_error`

Exceptions:

`SSE` is a variable:

`instantiation_error`

`SSE` is neither a variable nor an open opaque SSE handle:

`domain_error(http_sse_handle,SSE)`

`SSE` is an open opaque SSE handle but refers to a closed opaque SSE handle:

`existence_error(http_sse_handle,SSE)`

A field line uses a line ending other than a line feed or a carriage return immediately followed by a line feed:

`domain_error(http_sse_line_ending,Byte)`

A field line exceeds the configured maximum length:

`domain_error(http_sse_field_length,Length)`

---

receive/3

Reads the next SSE event using the opaque handle and the given read options. The direct API accepts a `max_field_length(Bytes|none)` override of the handle default set by `open/3` or `accept/4`.

Compilation flags:
`static`

Template:
`receive(SSE,Event,Options)`
Mode and number of proofs:
`receive(+compound,--term,+list)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
`Options` contains an invalid direct SSE option:
`domain_error(http_sse_option,Option)`
A field line uses a line ending other than a line feed or a carriage return immediately followed by a line feed:
`domain_error(http_sse_line_ending,Byte)`
A field line exceeds the configured maximum length:
`domain_error(http_sse_field_length,Length)`

---

close/1

Best-effort close of the SSE handle.

Compilation flags:
`static`

Template:
`close(SSE)`
Mode and number of proofs:
`close(+compound)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`SSE` refers to a closed opaque SSE handle:

existence_error(http_sse_handle,SSE)

---

close/2

Best-effort close of the SSE handle. First attempts to write the given final record, which must be valid for send/2; the handle is closed and deregistered regardless of whether that write succeeds.

Compilation flags:

static

Template:

close(SSE,Event)

Mode and number of proofs:

close(+compound,+term) - one_or_error

Exceptions:

SSE is a variable:

instantiation_error

SSE is neither a variable nor an open opaque SSE handle:

domain_error(http_sse_handle,SSE)

Event is not a valid outbound SSE record:

domain_error(http_sse_event,Event)

---

property/2

Enumerates properties of an open opaque SSE handle. Supported properties are role(client|server), transport(Transport), response(Response), connection(Connection), client_info(ClientInfo) (server-side handles only), last_event_id(Id|none), and retry(Milliseconds|none).

Compilation flags:

static

Template:

property(SSE,Property)

Mode and number of proofs:

property(+compound,+compound) - zero_or_one

property(+compound,--compound) - zero_or_more

---

send_data/2

Convenience predicate equivalent to `send(SSE, data(Data))`.

Compilation flags:
`static`

Template:
`send_data(SSE,Data)`
Mode and number of proofs:
`send_data(+compound,+atom)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`SSE` is an open opaque SSE handle but refers to a closed opaque SSE handle:
`existence_error(http_sse_handle,SSE)`
`data(Data)` is not a valid outbound SSE record:
`domain_error(http_sse_event,Event)`

---

send_event/3

Convenience predicate equivalent to `send(SSE, event(Type, Data))`.

Compilation flags:
`static`

Template:
`send_event(SSE,Type,Data)`
Mode and number of proofs:
`send_event(+compound,+atom,+atom)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`SSE` is an open opaque SSE handle but refers to a closed opaque SSE handle:
`existence_error(http_sse_handle,SSE)`
`event(Type, Data)` is not a valid outbound SSE record:
`domain_error(http_sse_event,Event)`

---

send_id_data/3

Convenience predicate equivalent to `send(SSE, id_data(Id, Data))`.

Compilation flags:
`static`

Template:
`send_id_data(SSE,Id,Data)`
Mode and number of proofs:
`send_id_data(+compound,+atom,+atom)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`SSE` is an open opaque SSE handle but refers to a closed opaque SSE handle:
`existence_error(http_sse_handle,SSE)`
`send(SSE, id_data(Id, Data))` is not a valid outbound SSE record:
`domain_error(http_sse_event,Event)`

---

send_comment/2

Convenience predicate equivalent to `send(SSE, comment(Text))`. Commonly used to send an idle keep-alive ping that the peer silently discards.

Compilation flags:
`static`

Template:
`send_comment(SSE,Text)`
Mode and number of proofs:
`send_comment(+compound,+atom)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`SSE` is an open opaque SSE handle but refers to a closed opaque SSE handle:
`existence_error(http_sse_handle,SSE)`
`send(SSE, comment(Text))` is not a valid outbound SSE record:
`domain_error(http_sse_event,Event)`

---

send_retry/2

Convenience predicate equivalent to `send(SSE, retry(Milliseconds))`.

Compilation flags:
`static`

Template:
`send_retry(SSE,Milliseconds)`
Mode and number of proofs:
`send_retry(+compound,+integer)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`SSE` is an open opaque SSE handle but refers to a closed opaque SSE handle:
`existence_error(http_sse_handle,SSE)`
`send(SSE, retry(Milliseconds))` is not a valid outbound SSE record:
`domain_error(http_sse_event,Event)`

---

send_json/2

Convenience predicate that JSON-encodes `JSON` and sends it as a data-only event.

Compilation flags:
`static`

Template:
`send_json(SSE,JSON)`
Mode and number of proofs:
`send_json(+compound,+term)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`SSE` refers to a closed opaque SSE handle:
`existence_error(http_sse_handle,SSE)`

---

receive_json/2

Convenience predicate that reads the next dispatched event and JSON-decodes its data.

Compilation flags:
`static`

Template:
`receive_json(SSE,JSON)`
Mode and number of proofs:
`receive_json(+compound,--term)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`SSE` is an open opaque SSE handle but refers to a closed opaque SSE handle:
`existence_error(http_sse_handle,SSE)`
The event data is not valid JSON text:
`domain_error(http_sse_json_text,Text)`

---

send_term/2

Convenience predicate that writes `Term` to an atom and sends it as a data-only event.

Compilation flags:
`static`

Template:
`send_term(SSE,Term)`
Mode and number of proofs:
`send_term(+compound,+term)` - `one_or_error`

Exceptions:
`SSE` is a variable:
`instantiation_error`
`SSE` is neither a variable nor an open opaque SSE handle:
`domain_error(http_sse_handle,SSE)`
`SSE` is an open opaque SSE handle but refers to a closed opaque SSE handle:
`existence_error(http_sse_handle,SSE)`
`term(Term)` is not a valid outbound SSE record:
`domain_error(http_sse_event,Event)`

---

receive_term/2

Convenience predicate that reads the next dispatched event and reads its data back as a term.

Compilation flags:
`static`

Template:
`receive_term(SSE,Term)`
Mode and number of proofs:
`receive_term(+compound,--term)` - `one_or_error`

Exceptions:
The event data does not read back as a term:
`domain_error(http_sse_term_text,Text)`

---

receive_data/2

Convenience predicate that reads the next dispatched event and returns just its data.

Compilation flags:
`static`

Template:
`receive_data(SSE,Data)`
Mode and number of proofs:
`receive_data(+compound,--atom)` - `one_or_error`

Exceptions:
The received record is not a dispatched event:
`domain_error(http_sse_event_message,Event)`

---

open_session/4

Convenience wrapper that opens a client SSE connection and runs one callback-driven client session, reconnecting with the last received event id when the connection drops, until the handler signals `stop` or reconnection is exhausted. Equivalent to `open_session/5` with an empty options list.

Compilation flags:
`static`

Template:
`open_session(URL,Handler,Response,State)`
Mode and number of proofs:
`open_session(+atom,+object_identifier,--compound,--compound)` - `one_or_error`

Exceptions:

`URL` is a variable:

```
instantiation_error
```

`URL` is neither a variable nor a supported absolute SSE URL:

```
domain_error(http_client_sse_url,URL)
```

`URL` is a supported absolute SSE URL but uses an unsupported SSE scheme:

```
domain_error(http_client_sse_scheme,Scheme)
```

`Handler` is a variable:

```
instantiation_error
```

`Handler` is neither a variable nor an existing object:

```
existence_error(object,Handler)
```

`Handler` does not conform to `http_sse_service_handler_protocol`:

```
domain_error(http_sse_service_handler,Handler)
```

`URL` is not a supported absolute SSE URL:

```
domain_error(http_client_sse_url,URL)
```

`URL` uses an unsupported SSE scheme:

```
domain_error(http_client_sse_scheme,Scheme)
```

The SSE server response is not a `200 text/event-stream` response:

```
domain_error(http_client_sse_response,Response)
```

---

open_session/5

Convenience wrapper over `open_session/4` that also accepts `reconnect(on|off)` (default `on`) and `max_reconnect_attempts(Count|infinite)` (default `infinite`) session loop options, in addition to the direct client options accepted by `open/3`. Reconnection is attempted immediately, without an inter-attempt delay. `State` is unified with `closed(LastEventId)` when the session ends.

Compilation flags:

```
static
```

Template:

```
open_session(URL,Handler,Response,State,Options)
```

Mode and number of proofs:

```
open_session(+atom,+object_identifier,--compound,--compound,+list) - one_or_error
```

Exceptions:

`URL` is a variable:

```
instantiation_error
```

`URL` is neither a variable nor a supported absolute SSE URL:

```
domain_error(http_client_sse_url,URL)
```

`URL` is a supported absolute SSE URL but uses an unsupported SSE scheme:

```
domain_error(http_client_sse_scheme,Scheme)
```

`Handler` is a variable:

```
instantiation_error
```

`Handler` is neither a variable nor an existing object:

```
existence_error(object,Handler)
```

`Handler` is an object but does not conform to `http_sse_service_handler_protocol`:

`domain_error(http_sse_service_handler,Handler)`
`Options` contains an invalid SSE service loop option:
`domain_error(http_sse_service_option,Option)`

---

serve_once/5

Convenience wrapper that accepts one incoming SSE request on the given listener using the default response policy and runs one callback-driven server session, repeatedly asking the handler for the next records to write until it signals `stop`, then closes the connection. Equivalent to `serve_once/6` with an empty options list.

Compilation flags:
`static`

Template:
`serve_once(Listener,Handler,Response,State,ClientInfo)`
Mode and number of proofs:
`serve_once(+compound,+object_identifier,--compound,--compound,--compound)` - `one_or_error`

Exceptions:
`Handler` is a variable:
`instantiation_error`
`Handler` is neither a variable nor an existing object:
`existence_error(object,Handler)`
`Handler` is an object but does not conform to `http_sse_service_handler_protocol`:
`domain_error(http_sse_service_handler,Handler)`
The SSE opening request does not exist:
`existence_error(http_socket_transport_sse_request,end_of_file)`
The SSE opening response is invalid:
`domain_error(http_socket_transport_sse_response,Response)`
The delegated HTTP server rejects the response stream:
`domain_error(http_response_stream,Error)`

---

serve_once/6

Convenience wrapper that accepts one incoming SSE request on the given listener and runs one callback-driven server session using the given combined acceptance and direct options, as accepted by `accept/4`.

Compilation flags:
`static`

Template:
`serve_once(Listener,Handler,Response,State,ClientInfo,Options)`
Mode and number of proofs:
`serve_once(+compound,+object_identifier,--compound,--compound,--compound,+list)` - `one_or_error`

Exceptions:

`Handler` is a variable:

`instantiation_error`

`Handler` is neither a variable nor an existing object:

`existence_error(object,Handler)`

`Handler` is an object but does not conform to `http_sse_service_handler_protocol`:

`domain_error(http_sse_service_handler,Handler)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

handle_seed_/1

Last allocated opaque SSE handle identifier.

Compilation flags:

`dynamic`

Template:

`handle_seed_(HandleId)`

Mode and number of proofs:

`handle_seed_(?positive_integer)` - `zero_or_one`

---

handle_state_/9

Stored opaque SSE handle state.

Compilation flags:

`dynamic`

Template:

`handle_state_(HandleId,Role,Transport,Connection,Response,ClientInfo,LastEventId,Retry,MaxFieldLength)`

Mode and number of proofs:

`handle_state_(?positive_integer,?atom,?object_identifier,?compound,?compound,?term,?atom,?term,?term)` - `zero_or_more`

Operators

(none)

See also

http_sse_service_handler_protocol

object

1.140.2 `http_sse_accept_handler(Options)`

Internal response adapter used by the high-level http_sse server predicates.

Availability:
`logtalk_load(http_sse(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-27

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_handler_protocol*
Uses:
http_server_core

Remarks:
(none)

Inherited public predicates:
handle/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.140.3 `http_sse_service_handler_protocol`

Protocol implemented by callback-driven http_sse client and server session handlers.

Availability:
`logtalk_load(http_sse(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *handle/2*
  - *next/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

handle/2

Called by open_session/4-5 for each event dispatched on a client session, in arrival order. Continue must be unified with continue to keep listening for further events (reconnecting with the last received event id if the connection drops) or with stop to end the session.

Compilation flags:

```
static
```

Template:

```
handle(Event,Continue)
```

Mode and number of proofs:

```
handle(+compound,-atom) - one
```

---

next/4

Called by serve_once/5-6 to obtain the next batch of outbound SSE records to write on a server session. The first call is made with State0 bound to []. Messages must be a list of terms valid for send/2. Continue must be unified with continue to have Messages written and next/4 called again with the updated state, or with stop to have Messages written and the session and connection closed.

Compilation flags:

```
static
```

Template:

```
next(State0,State,Messages,Continue)
```

Mode and number of proofs:

```
next(+term,-term,-list,-atom) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.141 http_static_files

object

1.141.1 `http_static_files`

Router-agnostic static file response helper built on the normalized `http_core` library.

Availability:
`logtalk_load(http_static_files(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
`public` *http_docroot_paths*

Uses:
atom
date
http_core
http_cors
http_directory_listing
list
mime_types
os
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *serve/4*
  - *serve/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

serve/4

Serves a relative request path from the given document root using the default options and returns a normalized response.

Compilation flags:
`static`

Template:
`serve(Path,Request,DocumentRoot,Response)`
Mode and number of proofs:
`serve(+atom,+compound,+atom,-compound)` - `one_or_error`

Exceptions:
`Path` is not a safe document-root relative path:
`domain_error(http_docroot_relative_path,Path)`
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
`DocumentRoot` is not a valid absolute document root path:
`domain_error(http_docroot_document_root,DocumentRoot)`
`DocumentRoot` is not an existing static-files document root:
`domain_error(http_static_files_document_root,DocumentRoot)`
The generated response violates normalized HTTP response semantics:
`domain_error(http_header_semantics,Header)`

---

serve/5

Serves a relative request path from the given document root using the given options and returns a normalized response.

Compilation flags:
`static`

Template:
`serve(Path,Request,DocumentRoot,Options,Response)`
Mode and number of proofs:
`serve(+atom,+compound,+atom,+list(compound),-compound)` - `one_or_error`

Exceptions:
`Path` is not a safe document-root relative path:
`domain_error(http_docroot_relative_path,Path)`
`Request` is not a valid normalized HTTP request term:
`domain_error(http_request,Request)`
`DocumentRoot` is not a valid absolute document root path:
`domain_error(http_docroot_document_root,DocumentRoot)`
`DocumentRoot` is not an existing static-files document root:
`domain_error(http_static_files_document_root,DocumentRoot)`
`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The generated response violates normalized HTTP response semantics:

`domain_error(http_header_semantics,Header)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.142 http_websocket

object

1.142.1 `http_websocket`

High-level WebSocket predicates for opening and closing connections, exchanging messages, and running common client and server session loops.

Availability:

`logtalk_load(http_websocket(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *options*

Uses:

http_client
http_core
http_websocket_client_service(HTTPTransport)
http_websocket_client_session

http_websocket_messages
http_websocket_server_service(HTTPTransport)
http_websocket_server_session
json
list
term_io
url(Representation)

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *open/2*
  - *open/3*
  - *accept/3*
  - *accept/4*
  - *send/2*
  - *send/3*
  - *receive/2*
  - *receive/3*
  - *close/1*
  - *close/2*
  - *property/2*
  - *send_text/2*
  - *receive_text/2*
  - *send_binary/2*
  - *receive_binary/2*
  - *send_json/2*
  - *receive_json/2*
  - *send_term/2*
  - *receive_term/2*
  - *open_session/4*
  - *open_session/5*
  - *serve_once/5*
  - *serve_once/6*
- *Protected predicates*
- *Private predicates*
  - *handle_seed_/1*
  - *handle_state_/9*
- *Operators*

Public predicates

open/2

Opens a client WebSocket connection to the given `ws://` or `wss://` URL and returns an opaque handle managed by this object.

Compilation flags:
`static`

Template:
`open(URL,WebSocket)`
Mode and number of proofs:
`open(+atom,-compound)` - `one_or_error`

Exceptions:
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute WebSocket URL:
`domain_error(http_client_websocket_url,URL)`
`URL` uses an unsupported WebSocket scheme:
`domain_error(http_client_websocket_scheme,Scheme)`

---

open/3

Opens a client WebSocket connection to the given URL, returning an opaque handle. Client handshake options are forwarded to `http_client::open_websocket/4`. The direct API also accepts `auto_pong(on|off)` and `max_payload_length(Bytes|none)` options.

Compilation flags:
`static`

Template:
`open(URL,WebSocket,Options)`
Mode and number of proofs:
`open(+atom,-compound,+list)` - `one_or_error`

Exceptions:
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute WebSocket URL:
`domain_error(http_client_websocket_url,URL)`
`URL` uses an unsupported WebSocket scheme:
`domain_error(http_client_websocket_scheme,Scheme)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
`Options` contains an invalid direct WebSocket option:

`domain_error(http_websocket_option,Option)`

`Options` contains an invalid WebSocket client option:

`domain_error(http_client_websocket_option,Option)`

The WebSocket server response is not a valid opening handshake response:

`domain_error(http_client_websocket_response,Response)`

---

accept/3

Accepts one incoming WebSocket connection on the given listener using the default opening-handshake policy and returns an opaque server-side handle together with the accepted client information.

Compilation flags:

`static`

Template:

`accept(Listener,WebSocket,ClientInfo)`

Mode and number of proofs:

`accept(+compound,-compound,-compound)` - `one_or_error`

Exceptions:

The WebSocket opening request does not exist:

`existence_error(http_socket_transport_websocket_request,Request)`

The WebSocket opening response is invalid:

`domain_error(http_socket_transport_websocket_response,Response)`

---

accept/4

Accepts one incoming WebSocket connection on the given listener and returns an opaque server-side handle. Server opening-handshake options are forwarded to `http_server_core::accept_websocket/3`. The direct API also accepts `auto_pong(on|off)` and `max_payload_length(Bytes|none)` options.

Compilation flags:

`static`

Template:

`accept(Listener,WebSocket,ClientInfo,Options)`

Mode and number of proofs:

`accept(+compound,-compound,-compound,+list)` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

`Options` contains an invalid direct WebSocket option:

`domain_error(http_websocket_option,Option)`

The WebSocket opening request does not exist:

`existence_error(http_socket_transport_websocket_request,Request)`

The WebSocket opening response is invalid:

`domain_error(http_socket_transport_websocket_response,Response)`

---

send/2

Writes one outbound WebSocket message using the opaque handle. Accepts normalized `message(Type, Payload)` terms and the convenience wrappers `text(Text)`, `binary(Bytes)`, `ping(Bytes)`, `pong(Bytes)`, `close(Payload)`, `json(JSON)`, and `term(Term)`.

Compilation flags:

`static`

Template:

`send(WebSocket,Message)`

Mode and number of proofs:

`send(+compound,+term)` - `one_or_error`

Exceptions:

`WebSocket` is a variable:

`instantiation_error`

`WebSocket` is not an open opaque WebSocket handle:

`domain_error(http_websocket_handle,WebSocket)`

`WebSocket` refers to a closed opaque WebSocket handle:

`existence_error(http_websocket_handle,WebSocket)`

`Message` is not a valid outbound WebSocket message:

`domain_error(http_websocket_message,Message)`

---

send/3

Writes one outbound WebSocket message using the opaque handle and the given write options, forwarding those options to the underlying session `write_message/5` predicate.

Compilation flags:

`static`

Template:

`send(WebSocket,Message,Options)`

Mode and number of proofs:

`send(+compound,+term,+list)` - `one_or_error`

Exceptions:

`WebSocket` is a variable:

instantiation_error
WebSocket is not an open opaque WebSocket handle:
domain_error(http_websocket_handle,WebSocket)
Message is not a valid outbound WebSocket message:
domain_error(http_websocket_message,Message)
Options is a variable or a partial list:
instantiation_error
Options is neither a variable nor a list:
type_error(list,Options)
Options contains an invalid write option:
domain_error(http_websocket_session_write_option,Option)

---

receive/2

Reads the next normalized WebSocket message using the opaque handle. Returns end_of_file when the peer closes the transport before another message is available.

Compilation flags:
static

Template:
receive(WebSocket,Message)
Mode and number of proofs:
receive(+compound,-term) - one_or_error

Exceptions:
WebSocket is a variable:
instantiation_error
WebSocket is not an open opaque WebSocket handle:
domain_error(http_websocket_handle,WebSocket)
WebSocket refers to a closed opaque WebSocket handle:
existence_error(http_websocket_handle,WebSocket)
The inbound WebSocket session sequence is invalid:
domain_error(http_websocket_session_sequence,Frame)

---

receive/3

Reads the next normalized WebSocket message using the opaque handle and the given read options. The direct API accepts per-call auto_pong(on|off) and max_payload_length(Bytes|none) overrides.

Compilation flags:
static

Template:
receive(WebSocket,Message,Options)

Mode and number of proofs:

`receive(+compound,-term,+list)` - `one_or_error`

Exceptions:

`WebSocket` is a variable:

`instantiation_error`

`WebSocket` is not an open opaque WebSocket handle:

`domain_error(http_websocket_handle,WebSocket)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

`Options` contains an invalid direct WebSocket option:

`domain_error(http_websocket_option,Option)`

The inbound WebSocket session sequence is invalid:

`domain_error(http_websocket_session_sequence,Frame)`

`close/1`

Best-effort graceful close of the WebSocket handle using the normal close payload `empty`.

Compilation flags:

`static`

Template:

`close(WebSocket)`

Mode and number of proofs:

`close(+compound)` - `one_or_error`

Exceptions:

`WebSocket` is a variable:

`instantiation_error`

`WebSocket` is not an open opaque WebSocket handle:

`domain_error(http_websocket_handle,WebSocket)`

`WebSocket` refers to a closed opaque WebSocket handle:

`existence_error(http_websocket_handle,WebSocket)`

---

close/2

Best-effort graceful close of the WebSocket handle using the given close payload, which must be valid for a normalized `message(close, Payload)` term.

Compilation flags:
`static`

Template:
`close(WebSocket,Payload)`

Mode and number of proofs:
`close(+compound,+term)` - `one_or_error`

Exceptions:
`WebSocket` is a variable:
`instantiation_error`
`WebSocket` is not an open opaque WebSocket handle:
`domain_error(http_websocket_handle,WebSocket)`
`Payload` is not a valid close payload:
`domain_error(http_websocket_close_payload,Payload)`

---

property/2

Enumerates properties of an open opaque WebSocket handle. Supported properties are `role(Role)`, `transport(Transport)`, `response(Response)`, `connection(Connection)`, `state(State)`, `client_info(ClientInfo)`, and `subprotocol(Protocol)`.

Compilation flags:
`static`

Template:
`property(WebSocket,Property)`

Mode and number of proofs:
`property(+compound,+compound)` - `zero_or_one`
`property(+compound,-compound)` - `zero_or_more`

---

send_text/2

Convenience wrapper for sending one text message.

Compilation flags:
`static`

Template:
`send_text(WebSocket,Text)`

Mode and number of proofs:

`send_text(+compound,+atom)` - `one_or_error`

Exceptions:

`WebSocket` is not an open opaque WebSocket handle:

`domain_error(http_websocket_handle,WebSocket)`

`Text` is not a valid text payload:

`domain_error(http_websocket_message_text,Text)`

---

receive_text/2

Reads the next message and returns its text payload when the message type is `text`. Returns `end_of_file` unchanged.

Compilation flags:

`static`

Template:

`receive_text(WebSocket,Text)`

Mode and number of proofs:

`receive_text(+compound,-term)` - `one_or_error`

Exceptions:

`WebSocket` is not an open opaque WebSocket handle:

`domain_error(http_websocket_handle,WebSocket)`

The received message is not a text message:

`domain_error(http_websocket_text_message,Message)`

---

send_binary/2

Convenience wrapper for sending one binary message.

Compilation flags:

`static`

Template:

`send_binary(WebSocket,Bytes)`

Mode and number of proofs:

`send_binary(+compound,+list(byte))` - `one_or_error`

Exceptions:

`WebSocket` is not an open opaque WebSocket handle:

`domain_error(http_websocket_handle,WebSocket)`

`Bytes` is not a valid binary payload:

`domain_error(http_websocket_message,Message)`

---

receive_binary/2

Reads the next message and returns its binary payload when the message type is `binary`. Returns `end_of_file` unchanged.

Compilation flags:
`static`

Template:
`receive_binary(WebSocket,Bytes)`
Mode and number of proofs:
`receive_binary(+compound,-term)` - `one_or_error`

Exceptions:
`WebSocket` is not an open opaque WebSocket handle:
`domain_error(http_websocket_handle,WebSocket)`
The received message is not a binary message:
`domain_error(http_websocket_binary_message,Message)`

---

send_json/2

Serializes a JSON term as UTF-8 text and sends it as one text message. Uses curly terms for parsed JSON objects, dashes for parsed JSON pairs, and atoms for parsed JSON strings.

Compilation flags:
`static`

Template:
`send_json(WebSocket,JSON)`
Mode and number of proofs:
`send_json(+compound,+nonvar)` - `one_or_error`

Exceptions:
`WebSocket` is not an open opaque WebSocket handle:
`domain_error(http_websocket_handle,WebSocket)`
`JSON` cannot be serialized as a text WebSocket message:
`domain_error(json_sink,JSON)`

---

receive_json/2

Reads the next text message and decodes its payload as JSON. Returns `end_of_file` unchanged. Uses curly terms for parsed JSON objects, dashes for parsed JSON pairs, and atoms for parsed JSON strings.

Compilation flags:
`static`

Template:

```
receive_json(WebSocket,JSON)
```

Mode and number of proofs:

```
receive_json(+compound,-term) - one_or_error
```

Exceptions:

`WebSocket` is not an open opaque WebSocket handle:

```
domain_error(http_websocket_handle,WebSocket)
```

The received text payload is not valid JSON:

```
domain_error(http_websocket_json_text,Text)
```

---

send_term/2

Serializes a Prolog term using canonical write options and sends it as one text message.

Compilation flags:

```
static
```

Template:

```
send_term(WebSocket,Term)
```

Mode and number of proofs:

```
send_term(+compound,+term) - one_or_error
```

Exceptions:

`WebSocket` is not an open opaque WebSocket handle:

```
domain_error(http_websocket_handle,WebSocket)
```

`Term` cannot be serialized as a text WebSocket message:

```
domain_error(http_websocket_term_text,Text)
```

---

receive_term/2

Reads the next text message and parses its payload as one Prolog term. Returns `end_of_file` unchanged.

Compilation flags:

```
static
```

Template:

```
receive_term(WebSocket,Term)
```

Mode and number of proofs:

```
receive_term(+compound,-term) - one_or_error
```

Exceptions:

`WebSocket` is not an open opaque WebSocket handle:

```
domain_error(http_websocket_handle,WebSocket)
```

The received text payload is not a canonical Prolog term:
`domain_error(http_websocket_term_text,Text)`

---

open_session/4

Convenience wrapper over `http_websocket_client_service::open/4` that keeps the user in the high-level `http_websocket` surface for callback-driven client sessions.

Compilation flags:
`static`

Template:
`open_session(URL,Handler,Response,State)`

Mode and number of proofs:
`open_session(+atom,+object_identifier,-compound,-compound)` - `one_or_error`

Exceptions:
`URL` is not a supported absolute WebSocket URL:
`domain_error(http_client_websocket_url,URL)`
`Handler` is not a valid WebSocket service handler:
`domain_error(http_websocket_service_handler,Handler)`

---

open_session/5

Convenience wrapper over `http_websocket_client_service::open/5`.

Compilation flags:
`static`

Template:
`open_session(URL,Handler,Response,State,Options)`

Mode and number of proofs:
`open_session(+atom,+object_identifier,-compound,-compound,+list)` - `one_or_error`

Exceptions:
`URL` is not a supported absolute WebSocket URL:
`domain_error(http_client_websocket_url,URL)`
`Handler` is not a valid WebSocket service handler:
`domain_error(http_websocket_service_handler,Handler)`
`Options` contains an invalid WebSocket client service option:
`domain_error(http_websocket_client_service_option,Option)`
`Options` contains an invalid WebSocket service loop option:
`domain_error(http_websocket_service_option,Option)`

---

serve_once/5

Convenience wrapper that accepts one incoming WebSocket connection on the given listener, performs the opening handshake using the default server policy, and runs one callback-driven server session.

Compilation flags:

`static`

Template:

`serve_once(Listener,Handler,Response,State,ClientInfo)`

Mode and number of proofs:

`serve_once(+compound,+object_identifier,-compound,-compound,-compound)` - `one_or_error`

Exceptions:

`Handler` is not a valid WebSocket service handler:

`domain_error(http_websocket_service_handler,Handler)`

The WebSocket opening request does not exist:

`existence_error(http_socket_transport_websocket_request,Request)`

The WebSocket opening response is invalid:

`domain_error(http_socket_transport_websocket_response,Response)`

---

serve_once/6

Convenience wrapper that accepts one incoming WebSocket connection on the given listener and runs one callback-driven server session using the given combined handshake and session options.

Compilation flags:

`static`

Template:

`serve_once(Listener,Handler,Response,State,ClientInfo,Options)`

Mode and number of proofs:

`serve_once(+compound,+object_identifier,-compound,-compound,-compound,+list)` - `one_or_error`

Exceptions:

`Handler` is not a valid WebSocket service handler:

`domain_error(http_websocket_service_handler,Handler)`

`Options` contains an invalid WebSocket service loop option:

`domain_error(http_websocket_service_option,Option)`

The WebSocket opening request does not exist:

`existence_error(http_socket_transport_websocket_request,Request)`

The WebSocket opening response is invalid:

`domain_error(http_socket_transport_websocket_response,Response)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

handle_seed_/1

Last allocated opaque WebSocket handle identifier.

Compilation flags:
```
dynamic
```

Template:
```
handle_seed_(HandleId)
```
Mode and number of proofs:
```
handle_seed_(?positive_integer) - zero_or_one
```

---

handle_state_/9

Stored opaque WebSocket handle state.

Compilation flags:
```
dynamic
```

Template:
```
handle_state_(HandleId,Role,Transport,Connection,Response,ClientInfo,State,
AutoPong,MaxPayloadLength)
```
Mode and number of proofs:
```
handle_state_(?positive_integer,?atom,?object_identifier,?compound,?compound,?
term,?compound,?atom,?term) - zero_or_more
```

Operators

(none)
object

1.142.2 http_websocket_accept_handler(Options)

Internal opening-handshake adapter used by the high-level http_websocket server predicates.

Availability:
```
logtalk_load(http_websocket(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_handler_protocol*
Uses:
http_server_core

Remarks:
(none)

Inherited public predicates:
handle/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.143 http_websocket_frames

object

1.143.1 `http_websocket_frames`

Transport-neutral WebSocket frame predicates for constructing, parsing, generating, reading, and writing normalized frame terms.

Availability:
`logtalk_load(http_websocket_frames(loader))`

Author: Paulo Moura
Version: 1:1:0

Date: 2026-10-22

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*

Uses:
list
reader

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *frame/5*
  - *is_frame/1*
  - *parse/2*
  - *generate/2*
  - *read_frame/2*
  - *read_frame/3*
  - *write_frame/2*
  - *final/2*
  - *opcode/2*
  - *payload/2*
  - *properties/2*
  - *property/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

frame/5

Constructs a validated normalized WebSocket frame term from the fragmentation state atom, opcode atom, payload bytes, and frame properties.

Compilation flags:
`static`

Template:
`frame(Final,Opcode,Payload,Properties,Frame)`

Mode and number of proofs:
`frame(+atom,+atom,+list(byte),+list(compound),-compound)` - `one_or_error`

Exceptions:

`Final` is not a valid WebSocket final flag:
`domain_error(http_websocket_final,Final)`
`Opcode` is not a valid WebSocket opcode:
`domain_error(http_websocket_opcode,Opcode)`
`Payload` is a variable or a partial list:
`instantiation_error`
`Payload` is not a valid WebSocket payload byte list:
`domain_error(http_websocket_payload,Payload)`
`Properties` is a variable or a partial list:
`instantiation_error`
`Properties` is not a valid WebSocket frame property list:
`domain_error(http_websocket_properties,Properties)`
A frame property is invalid:
`domain_error(http_websocket_property,Property)`
A masking key is invalid:
`domain_error(http_websocket_masking_key,Key)`
Reserved bits are invalid:
`domain_error(http_websocket_reserved_bits,Bits)`
The frame violates WebSocket frame semantics:
`domain_error(http_websocket_frame,Frame)`

Remarks:

- Final atoms: Use `final` for FIN=1 and `more` for FIN=0.
- Opcode atoms: Supported opcodes are `continuation`, `text`, `binary`, `close`, `ping`, and `pong`.
- Property `masking_key(Key)`: Requests masking on the wire using a list of four mask bytes.
- Property `reserved_bits(Bits)`: Preserves reserved bits as an ordered list containing any of `rsv1`, `rsv2`, and `rsv3`.

`is_frame/1`

True when the argument is a valid normalized WebSocket frame term.

Compilation flags:
`static`

Template:
`is_frame(Frame)`
Mode and number of proofs:
`is_frame(@term)` - `zero_or_one`

---

parse/2

Parses one WebSocket frame from a source term. Supported sources are `bytes(Bytes)`, `codes(Bytes)`, `file(File)`, and `stream(Stream)`. Empty sources return `end_of_file`.

Compilation flags:
`static`

Template:
`parse(Source,Frame)`
Mode and number of proofs:
`parse(++compound,-term)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid WebSocket frame source term:
`domain_error(http_websocket_source,Source)`
The source bytes are not a valid WebSocket byte sequence:
`domain_error(http_websocket_byte_sequence,Bytes)`
The source bytes contain an invalid WebSocket opcode:
`domain_error(http_websocket_opcode,Opcode)`
The source bytes contain an invalid WebSocket frame length:
`domain_error(http_websocket_frame_length,PayloadLength)`
The source bytes do not encode a valid normalized WebSocket frame:
`domain_error(http_websocket_frame,Frame)`

---

generate/2

Generates one validated WebSocket frame to a sink term. Supported sinks are `bytes(Bytes)`, `codes(Bytes)`, `file(File)`, and `stream(Stream)`.

Compilation flags:
`static`

Template:
`generate(Sink,Frame)`
Mode and number of proofs:
`generate(++compound,+compound)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid WebSocket frame sink term:
`domain_error(http_websocket_sink,Sink)`
`Frame` is not a valid normalized WebSocket frame term:
`domain_error(http_websocket_frame,Frame)`

---

read_frame/2

Reads one WebSocket frame from a binary stream without requiring end-of-file. Returns `end_of_file` when the stream is already exhausted.

Compilation flags:

`static`

Template:

`read_frame(Stream,Frame)`

Mode and number of proofs:

`read_frame(+stream_or_alias,-term)` - `one_or_error`

Exceptions:

`Stream` is a variable:

`instantiation_error`

The inbound bytes are not a valid WebSocket byte sequence:

`domain_error(http_websocket_byte_sequence,Bytes)`

The inbound payload length exceeds the configured limit:

`domain_error(http_websocket_payload_length_limit,PayloadLength)`

The inbound bytes do not encode a valid normalized WebSocket frame:

`domain_error(http_websocket_frame,Frame)`

---

read_frame/3

Reads one WebSocket frame from a binary stream using the given read options. Returns `end_of_file` when the stream is already exhausted.

Compilation flags:

`static`

Template:

`read_frame(Stream,Frame,Options)`

Mode and number of proofs:

`read_frame(+stream_or_alias,-term,+list)` - `one_or_error`

Exceptions:

`Stream` is a variable:

`instantiation_error`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The inbound payload length exceeds the configured limit:
`domain_error(http_websocket_payload_length_limit,PayloadLength)`
The inbound bytes are not a valid WebSocket byte sequence:
`domain_error(http_websocket_byte_sequence,Bytes)`
The inbound bytes do not encode a valid normalized WebSocket frame:
`domain_error(http_websocket_frame,Frame)`

Remarks:

- Repeated options: When the same read option is given multiple times, the first occurrence is used.
- Option `max_payload_length(Bytes)`: Rejects frames whose declared payload length is greater than `Bytes` before allocating payload storage. Use a non-negative integer.

---

write_frame/2

Writes one validated WebSocket frame to a binary stream and flushes the stream.

Compilation flags:
`static`

Template:
`write_frame(Stream,Frame)`
Mode and number of proofs:
`write_frame(+stream_or_alias,+compound)` - `one_or_error`

Exceptions:
`Stream` is a variable:
`instantiation_error`
`Frame` is not a valid normalized WebSocket frame term:
`domain_error(http_websocket_frame,Frame)`

---

final/2

Returns the fragmentation state atom of a validated WebSocket frame term.

Compilation flags:
`static`

Template:
`final(Frame,Final)`
Mode and number of proofs:
`final(+compound,-atom)` - `one_or_error`

Exceptions:
`Frame` is not a valid normalized WebSocket frame term:
`domain_error(http_websocket_frame,Frame)`

---

opcode/2

Returns the opcode atom of a validated WebSocket frame term.

Compilation flags:
`static`

Template:
`opcode(Frame,Opcode)`
Mode and number of proofs:
`opcode(+compound,-atom)` - `one_or_error`

Exceptions:
`Frame` is not a valid normalized WebSocket frame term:
`domain_error(http_websocket_frame,Frame)`

---

payload/2

Returns the payload byte list of a validated WebSocket frame term.

Compilation flags:
`static`

Template:
`payload(Frame,Payload)`
Mode and number of proofs:
`payload(+compound,-list(byte))` - `one_or_error`

Exceptions:
`Frame` is not a valid normalized WebSocket frame term:
`domain_error(http_websocket_frame,Frame)`

---

properties/2

Returns the normalized property list of a validated WebSocket frame term.

Compilation flags:
`static`

Template:
`properties(Frame,Properties)`
Mode and number of proofs:
`properties(+compound,-list(compound))` - `one_or_error`

Exceptions:

`Frame` is not a valid normalized WebSocket frame term:
`domain_error(http_websocket_frame,Frame)`

`property/2`

True when the validated WebSocket frame term contains the given property. When the property argument is a variable, properties can be enumerated on backtracking.

Compilation flags:
`static`

Template:
`property(Frame,Property)`
Mode and number of proofs:
`property(+compound,?compound)` - `zero_or_one_or_error`

Exceptions:
`Frame` is not a valid normalized WebSocket frame term:
`domain_error(http_websocket_frame,Frame)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.144 http_websocket_handshake

object

1.144.1 `http_websocket_handshake`

Shared helpers for WebSocket opening-handshake key generation and accept-value computation.

Availability:
`logtalk_load(http_websocket_handshake(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-04-08

Compilation flags:
`static, context_switching_calls`

Uses:
base64
byte_order
crypto
hash_common_32
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *websocket_opening_key/1*
  - *websocket_accept/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

websocket_opening_key/1

Generates a canonical base64-encoded 16-byte WebSocket opening key.

Compilation flags:
`static`

Template:
`websocket_opening_key(Key)`

Mode and number of proofs:
`websocket_opening_key(--atom)` - `one`

`websocket_accept/2`

Computes the canonical *Sec-WebSocket-Accept* value for a valid WebSocket opening key.

Compilation flags:
`static`

Template:
`websocket_accept(Key,Accept)`
Mode and number of proofs:
`websocket_accept(++term,--atom)` - `one_or_error`

Exceptions:
`Key` is a variable:
`instantiation_error`
`Key` is not a valid WebSocket opening key value:
`domain_error(http_websocket_key,Key)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.145 http_websocket_messages

object

1.145.1 `http_websocket_messages`

Transport-neutral WebSocket message predicates using atoms for text messages and close reasons.

Availability:
`logtalk_load(http_websocket_messages(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *http_websocket_messages(atom)*

Remarks:
(none)

Inherited public predicates:
is_message/1 *message/3* *payload/2* *read_message/2* *type/2* *write_message/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.145.2 `http_websocket_messages(TextRepresentation)`

- `TextRepresentation` - Text representation to be used for text messages and close reasons. Possible values are `atom` (default), `chars`, and `codes`.

Transport-neutral WebSocket message predicates built on top of the http_websocket frame layer.

Availability:
`logtalk_load(http_websocket_messages(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static, context_switching_calls`

Uses:
http_websocket_frames

list
utf_8

Remarks:

- Fragmentation model: This layer reassembles fragmented text and binary messages only when all continuation frames arrive contiguously. Interleaved control frames during fragmented messages raise `http_websocket_message_sequence`. Use `http_websocket_session` for RFC 6455 live-stream processing that must surface interleaved control frames while reassembling data messages.

Inherited public predicates:
(none)

- *Public predicates*
  - *message/3*
  - *is_message/1*
  - *read_message/2*
  - *write_message/2*
  - *type/2*
  - *payload/2*
- *Protected predicates*
  - *encode_message/4*
  - *decode_message/3*
- *Private predicates*
- *Operators*

Public predicates

message/3

Constructs a validated normalized WebSocket message term from the message type atom and payload term.

Compilation flags:
`static`

Template:
`message(Type,Payload,Message)`

Mode and number of proofs:
`message(+atom,+term,-compound)` - `one_or_error`

Exceptions:
`Type` is not a supported WebSocket message type:
`domain_error(http_websocket_message_type,Type)`
`Payload` is not valid for the requested WebSocket message type:
`domain_error(http_websocket_message,Message)`
`Payload` is not a valid WebSocket close payload:
`domain_error(http_websocket_close_payload,Payload)`
`Payload` contains an invalid WebSocket close code:

`domain_error(http_websocket_close_code,Code)`

`Payload` is not valid text for the selected text representation:

`domain_error(http_websocket_message_text,Text)`

Remarks:

- Type `text`: Payload is text in the selected text representation.
- Type `binary`: Payload is a list of bytes.
- Type `ping` or `pong`: Payload is a list of bytes.
- Type `close`: Payload is one of `empty`, `status(Code)`, or `status(Code, Reason)`.

`is_message/1`

True when the argument is a valid normalized WebSocket message term.

Compilation flags:

`static`

Template:

`is_message(Message)`

Mode and number of proofs:

`is_message(@term)` - `zero_or_one`

`read_message/2`

Reads one WebSocket message from a binary stream. Continuation frames are reassembled for text and binary messages when they arrive contiguously. Returns `end_of_file` when the stream is already exhausted.

Compilation flags:

`static`

Template:

`read_message(Stream,Message)`

Mode and number of proofs:

`read_message(+stream_or_alias,-term)` - `one_or_error`

Exceptions:

`Stream` is a variable:

`instantiation_error`

The inbound frames do not form a valid normalized WebSocket message sequence:

`domain_error(http_websocket_message_sequence,Frame)`

The inbound frame is invalid:

`domain_error(http_websocket_frame,Frame)`

The inbound message payload is invalid:

`domain_error(http_websocket_message,Message)`

The inbound message text is invalid for the selected text representation:

```
domain_error(http_websocket_message_text,Text)
```

Remarks:

- Interleaved control frames: Control frames interleaved in the middle of a fragmented data message are not surfaced by this simplified layer and instead raise `http_websocket_message_sequence`. Use `http_websocket_session` when interleaved control frames must be handled while a fragmented message is in progress.

---

write_message/2

Writes one validated WebSocket message as a single final frame to a binary stream and flushes the stream.

Compilation flags:

```
static
```

Template:

```
write_message(Stream,Message)
```

Mode and number of proofs:

```
write_message(+stream_or_alias,+compound) - one_or_error
```

Exceptions:

`Stream` is a variable:

```
instantiation_error
```

`Message` is not a valid normalized WebSocket message term:

```
domain_error(http_websocket_message,Message)
```

`Message` has an unsupported message type:

```
domain_error(http_websocket_message_type,Type)
```

`Message` has an invalid close payload:

```
domain_error(http_websocket_close_payload,Payload)
```

`Message` has invalid text for the selected text representation:

```
domain_error(http_websocket_message_text,Text)
```

---

type/2

Returns the type atom of a validated WebSocket message term.

Compilation flags:

```
static
```

Template:

```
type(Message,Type)
```

Mode and number of proofs:

```
type(+compound,-atom) - one_or_error
```

Exceptions:

`Message` is not a valid normalized WebSocket message term:

`domain_error(http_websocket_message,Message)`
`Message` has an unsupported message type:
`domain_error(http_websocket_message_type,Type)`

---

payload/2

Returns the payload term of a validated WebSocket message term.

Compilation flags:
`static`

Template:
`payload(Message,Payload)`
Mode and number of proofs:
`payload(+compound,-term)` - `one_or_error`

Exceptions:
`Message` is not a valid normalized WebSocket message term:
`domain_error(http_websocket_message,Message)`
`Message` has an unsupported message type:
`domain_error(http_websocket_message_type,Type)`

Protected predicates

encode_message/4

Protected helper that normalizes a WebSocket message term and returns the normalized term together with the corresponding frame opcode and payload bytes.

Compilation flags:
`static`

Template:
`encode_message(Message,NormalizedMessage,Opcode,PayloadBytes)`
Mode and number of proofs:
`encode_message(+compound,-compound,-atom,-list(byte))` - `one_or_error`

Exceptions:
`Message` is not a valid normalized WebSocket message term:
`domain_error(http_websocket_message,Message)`
`Message` has an unsupported message type:
`domain_error(http_websocket_message_type,Type)`
`Message` has an invalid close payload:
`domain_error(http_websocket_close_payload,Payload)`
`Message` has invalid text for the selected text representation:

```
domain_error(http_websocket_message_text,Text)
```

---

decode_message/3

Protected helper that decodes payload bytes for the given message type atom into a normalized WebSocket message term.

Compilation flags:
`static`

Template:
`decode_message(Type,PayloadBytes,Message)`
Mode and number of proofs:
`decode_message(+atom,+list(byte),-compound)` - `one_or_error`

Exceptions:
`Type` is not a supported WebSocket message type:
`domain_error(http_websocket_message_type,Type)`
`PayloadBytes` is not a valid close payload:
`domain_error(http_websocket_close_payload,PayloadBytes)`
`PayloadBytes` contains an invalid close code:
`domain_error(http_websocket_close_code,Code)`
`PayloadBytes` does not decode to valid text for the selected text representation:
`domain_error(http_websocket_message_text,PayloadBytes)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.146 http_websocket_service

object

1.146.1 http_websocket_client_service

By default, the client-side convenience for callback-driven WebSocket sessions with atom text representation, combining the opening handshake, optional initial outbound messages, and the higher-level session loop, uses the `http_socket_transport` library.

Availability:
`logtalk_load(http_websocket_service(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *http_websocket_service(http_socket_transport,client,atom)*

Remarks:
(none)

Inherited public predicates:
initial_state/1 is_message/1 is_state/1 message/3 payload/2 read_message/2 read_message/4 read_message/5 read_message/6 run_session/3 run_session/4 type/2 write_message/2 write_message/3 write_message/4 write_message/5

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.146.2 `http_websocket_client_service(HTTPTransport)`

Client-side convenience for callback-driven WebSocket sessions with atom text representation, combining the opening handshake, optional initial outbound messages, and the higher-level session loop.

Availability:
`logtalk_load(http_websocket_service(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-13

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *http_websocket_service(HTTPTransport,client,atom)*

Uses:
http_client
list
user

Remarks:
(none)

Inherited public predicates:
initial_state/1 is_message/1 is_state/1 message/3 payload/2 read_message/2 read_message/4 read_message/5 read_message/6 run_session/3 run_session/4 type/2 write_message/2 write_message/3 write_message/4 write_message/5

- *Public predicates*
  - *open/4*
  - *open/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

open/4

Builds a WebSocket opening handshake from the given absolute WebSocket URL supported by the selected transport parameterization, opens the upgraded connection, optionally writes initial outbound messages, then runs one callback-driven client session loop until the close handshake completes or the peer closes the stream.

Compilation flags:
`static`

Template:
`open(URL,Handler,Response,State)`
Mode and number of proofs:
`open(+atom,+object_identifier,-compound,-compound)` - `one_or_error`

Exceptions:
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute WebSocket URL:
`domain_error(http_client_websocket_url,URL)`
`URL` uses an unsupported WebSocket scheme:
`domain_error(http_client_websocket_scheme,Scheme)`
The WebSocket server rejects the opening handshake version:

`domain_error(http_client_websocket_version_rejection,Response)`
The WebSocket server rejects authentication:
`domain_error(http_client_websocket_authentication_rejection,Response)`
The WebSocket server redirects the opening handshake:
`domain_error(http_client_websocket_redirection_rejection,Response)`
The WebSocket server rejects the opening handshake:
`domain_error(http_client_websocket_rejection,Response)`
The WebSocket server response is not a valid opening handshake response:
`domain_error(http_client_websocket_response,Response)`
`Handler` is a variable:
`instantiation_error`
`Handler` is not a valid WebSocket service handler:
`domain_error(http_websocket_service_handler,Handler)`
The upgraded connection handle is invalid:
`domain_error(http_socket_transport_connection,Connection)`
The delegated session loop raises a WebSocket session error:
`domain_error(http_websocket_session_sequence,Frame)`
`Handler` returns an invalid reply:
`domain_error(http_websocket_service_handler_reply,Reply)`

---

open/5

Builds a WebSocket opening handshake from the given absolute WebSocket URL supported by the selected transport parameterization, opens the upgraded connection, optionally writes initial outbound messages, then runs one callback-driven client session loop using the given combined handshake and session options.

Compilation flags:
`static`

Template:
`open(URL,Handler,Response,State,Options)`
Mode and number of proofs:
`open(+atom,+object_identifier,-compound,-compound,+list)` - `one_or_error`

Exceptions:
`URL` is a variable:
`instantiation_error`
`URL` is not a supported absolute WebSocket URL:
`domain_error(http_client_websocket_url,URL)`
`URL` uses an unsupported WebSocket scheme:
`domain_error(http_client_websocket_scheme,Scheme)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
`Options` contains an invalid initial message list:
`type_error(list,Messages)`

`Options` contains an invalid initial message:
`domain_error(http_websocket_client_service_initial_message,Message)`
`Options` contains an invalid automatic pong setting:
`domain_error(http_websocket_client_service_option,auto_pong(Option))`
`Options` contains an invalid client service option:
`domain_error(http_websocket_client_service_option,Option)`
`Options` contains an invalid WebSocket client option:
`domain_error(http_client_websocket_option,Option)`
`Options` contains an invalid WebSocket HTTP version:
`domain_error(http_client_websocket_version,Version)`
`Options` contains reserved WebSocket headers:
`domain_error(http_client_websocket_headers,Headers)`
`Options` contains an invalid WebSocket service loop option:
`domain_error(http_websocket_service_option,Option)`
Timed session-loop options are not available on this backend:
`existence_error(http_websocket_service,timing)`
The WebSocket server rejects the opening handshake version:
`domain_error(http_client_websocket_version_rejection,Response)`
The WebSocket server rejects authentication:
`domain_error(http_client_websocket_authentication_rejection,Response)`
The WebSocket server redirects the opening handshake:
`domain_error(http_client_websocket_redirection_rejection,Response)`
The WebSocket server rejects the opening handshake:
`domain_error(http_client_websocket_rejection,Response)`
The WebSocket server response is not a valid opening handshake response:
`domain_error(http_client_websocket_response,Response)`
`Handler` is a variable:
`instantiation_error`
`Handler` is not a valid WebSocket service handler:
`domain_error(http_websocket_service_handler,Handler)`
The upgraded connection handle is invalid:
`domain_error(http_socket_transport_connection,Connection)`
The delegated session loop raises a WebSocket session error:
`domain_error(http_websocket_session_sequence,Frame)`
`Handler` returns an invalid reply:
`domain_error(http_websocket_service_handler_reply,Reply)`

Remarks:

- Repeated options: When the same handshake or session option is given multiple times, the first occurrence is used.
- Handshake options: The *headers/1*, *query/1*, *version/1*, *protocols/1*, *key/1*, and *connection_options/1* options are forwarded to *open_websocket/4*. The accepted URL schemes depend on the selected transport parameterization, e.g. *ws://* for *http_socket_transport* and both *ws://* and *wss://* for *http_process_transport*.
- Session option `auto_pong(on|off)`: Controls automatic pong replies during the session loop.
- Session option `initial_messages(Messages)`: Writes the given list of normalized outbound messages before entering the session loop.
- Session option `keepalive_interval(Seconds)`: Schedules empty ping messages when the peer stays silent for the given positive number of seconds. This option requires backend thread support.
- Session option `idle_timeout(Seconds)`: Closes the session with `status(1001, idle_timeout)` after the given positive number of seconds without an inbound message. This option requires backend thread support.

- Session option `max_payload_length(Bytes)`: Rejects inbound frames whose declared payload length is greater than `Bytes` before allocating payload storage. Oversized frames are treated as `1009` close errors in the session loop. Use a non-negative integer.

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.146.3 `http_websocket_server_service`

By default, the server-side convenience for callback-driven WebSocket sessions with atom text representation, including registry-backed broadcast helpers, uses the `http_socket_transport` library.

Availability:
`logtalk_load(http_websocket_service(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *http_websocket_service(http_socket_transport,server,atom)*

Remarks:
(none)

Inherited public predicates:
initial_state/1 is_message/1 is_state/1 message/3 payload/2 read_message/2 read_message/4 read_message/5 read_message/6 run_session/3 run_session/4 type/2 write_message/2 write_message/3 write_message/4 write_message/5

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.146.4 `http_websocket_server_service(HTTPTransport)`

Server-side convenience for callback-driven WebSocket sessions with atom text representation, including registry-backed broadcast helpers.

Availability:

```
logtalk_load(http_websocket_service(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-13

Compilation flags:

```
static, context_switching_calls, threaded
```

Extends:

`public` *http_websocket_service(HTTPTransport,server,atom)*

Uses:

http_websocket_service_registry
user

Remarks:

(none)

Inherited public predicates:

initial_state/1 *is_message/1* *is_state/1* *message/3* *payload/2* *read_message/2* *read_message/4* *read_message/5* *read_message/6* *run_session/3* *run_session/4* *type/2* *write_message/2* *write_message/3* *write_message/4* *write_message/5*

- *Public predicates*
  - *serve_once/6*
  - *serve_once/7*
  - *serve_until_shutdown/5*

    - *serve_until_shutdown/6*
    - *request_shutdown/1*
- *Protected predicates*
- *Private predicates*
    - *shutdown_seed_/1*
    - *shutdown_control_/3*
    - *shutdown_requested_/2*
    - *active_worker_/3*
- *Operators*

Public predicates

serve_once/6

Accepts one incoming socket connection on the given listener created by the selected transport parameterization, serves one WebSocket opening handshake using the given HTTP handler, then runs one callback-driven WebSocket session using the given session handler until the close handshake completes or the peer closes the stream.

Compilation flags:

`static`

Template:

`serve_once(Listener,HandshakeHandler,SessionHandler,Response,State,ClientInfo)`

Mode and number of proofs:

`serve_once(+compound,+object_identifier,+object_identifier,-compound,-compound,-compound)` - `one_or_error`

Exceptions:

The listener is invalid for the selected transport:

`domain_error(http_socket_transport_listener,Listener)`

The WebSocket opening request does not exist:

`existence_error(http_socket_transport_websocket_request,end_of_file)`

The WebSocket opening response is invalid:

`domain_error(http_socket_transport_websocket_response,Response)`

`SessionHandler` is a variable:

`instantiation_error`

`SessionHandler` is not a valid WebSocket service handler:

`domain_error(http_websocket_service_handler,SessionHandler)`

The upgraded connection handle is invalid:

`domain_error(http_socket_transport_connection,Connection)`

The delegated session loop raises a WebSocket session error:

`domain_error(http_websocket_session_sequence,Frame)`

`SessionHandler` returns an invalid reply:

`domain_error(http_websocket_service_handler_reply,Reply)`

serve_once/7

Accepts one incoming socket connection on the given listener created by the selected transport parameterization, serves one WebSocket opening handshake using the given HTTP handler, then runs one callback-driven WebSocket session using the given session handler and the given session-loop options.

Compilation flags:

`static`

Template:

`serve_once(Listener,HandshakeHandler,SessionHandler,Response,State,ClientInfo,Options)`

Mode and number of proofs:

`serve_once(+compound,+object_identifier,+object_identifier,-compound,-compound,-compound,+list)` - `one_or_error`

Exceptions:

The listener is invalid for the selected transport:

`domain_error(http_socket_transport_listener,Listener)`

The WebSocket opening request does not exist:

`existence_error(http_socket_transport_websocket_request,end_of_file)`

The WebSocket opening response is invalid:

`domain_error(http_socket_transport_websocket_response,Response)`

`SessionHandler` is a variable:

`instantiation_error`

`SessionHandler` is not a valid WebSocket service handler:

`domain_error(http_websocket_service_handler,SessionHandler)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid WebSocket service loop option:

`domain_error(http_websocket_service_option,Option)`

Timed session-loop options are not available on this backend:

`existence_error(http_websocket_service,timing)`

The upgraded connection handle is invalid:

`domain_error(http_socket_transport_connection,Connection)`

The delegated session loop raises a WebSocket session error:

`domain_error(http_websocket_session_sequence,Frame)`

`SessionHandler` returns an invalid reply:

`domain_error(http_websocket_service_handler_reply,Reply)`

Remarks:

- Option `auto_pong(on)`: Automatically writes pong replies while still forwarding ping messages to the session handler.
- Option `auto_pong(off)`: Disables automatic pong replies. This is the default.

- Option `keepalive_interval(Seconds)`: Schedules empty ping messages when the peer stays silent for the given positive number of seconds. This option requires backend thread support.
- Option `idle_timeout(Seconds)`: Closes the session with `status(1001, idle_timeout)` after the given positive number of seconds without an inbound message. This option requires backend thread support.
- Option `max_payload_length(Bytes)`: Rejects inbound frames whose declared payload length is greater than `Bytes` before allocating payload storage. Oversized frames are treated as `1009` close errors in the session loop. Use a non-negative integer.

`serve_until_shutdown/5`

Accepts WebSocket opening handshakes on the given listener created by the selected transport parameterization until request_shutdown/1 is called for the specified control term, runs one callback-driven session loop per accepted upgraded connection, registers active sessions in the given registry for queued broadcasts, and closes the listener before returning.

Compilation flags:

`static`

Template:

`serve_until_shutdown(Listener,HandshakeHandler,SessionHandler,Registry,Control)`

Mode and number of proofs:

`serve_until_shutdown(+compound,+object_identifier,+object_identifier,+compound,+nonvar)` - `one_or_error`

Exceptions:

Registry-backed serving is not available on this backend:

`existence_error(http_websocket_server_service,registry)`

`Registry` is not an open WebSocket service registry handle:

`domain_error(http_websocket_service_registry,Registry)`

`Registry` refers to a closed WebSocket service registry handle:

```
existence_error(http_websocket_service_registry,
session_registry(RegistryId))
```

`Control` is a variable:

`instantiation_error`

`Control` is already registered for another open-ended WebSocket server loop:

`permission_error(reuse,http_websocket_service_shutdown_control,Control)`

The WebSocket opening response is invalid:

`domain_error(http_socket_transport_websocket_response,Response)`

`SessionHandler` is a variable:

`instantiation_error`

`SessionHandler` is not a valid WebSocket service handler:

`domain_error(http_websocket_service_handler,SessionHandler)`

The delegated session loop raises a WebSocket session error:

`domain_error(http_websocket_session_sequence,Frame)`

`SessionHandler` returns an invalid registry action:

`domain_error(http_websocket_service_handler_action,Action)`

Remarks:

- Thread support: This helper requires backend thread support so that multiple sessions can stay active concurrently.

- Handler actions: Plain normalized messages are written back to the originating session. `broadcast(Message)` queues `Message` for all registered sessions and `broadcast_others(Message)` queues it for all registered sessions except the originating one.

---

serve_until_shutdown/6

Accepts WebSocket opening handshakes on the given listener created by the selected transport parameterization until request_shutdown/1 is called for the specified control term, runs one registry-backed callback session per accepted connection, applies the given session-loop options to every active session, and closes the listener before returning.

Compilation flags:

`static`

Template:

`serve_until_shutdown(Listener,HandshakeHandler,SessionHandler,Registry,Control,Options)`

Mode and number of proofs:

`serve_until_shutdown(+compound,+object_identifier,+object_identifier,+compound,+nonvar,+list)` - `one_or_error`

Exceptions:

Registry-backed serving is not available on this backend:

`existence_error(http_websocket_server_service,registry)`

`Registry` is not an open WebSocket service registry handle:

`domain_error(http_websocket_service_registry,Registry)`

`Registry` refers to a closed WebSocket service registry handle:

`existence_error(http_websocket_service_registry,session_registry(RegistryId))`

`Control` is a variable:

`instantiation_error`

`Control` is already registered for another open-ended WebSocket server loop:

`permission_error(reuse,http_websocket_service_shutdown_control,Control)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

`Options` contains an invalid WebSocket service loop option:

`domain_error(http_websocket_service_option,Option)`

Timed session-loop options are not available on this backend:

`existence_error(http_websocket_service,timing)`

The WebSocket opening response is invalid:

`domain_error(http_socket_transport_websocket_response,Response)`

`SessionHandler` is a variable:

`instantiation_error`

`SessionHandler` is not a valid WebSocket service handler:

`domain_error(http_websocket_service_handler,SessionHandler)`

The delegated session loop raises a WebSocket session error:

`domain_error(http_websocket_session_sequence,Frame)`

`SessionHandler` returns an invalid registry action:

`domain_error(http_websocket_service_handler_action,Action)`

Remarks:

- Option `auto_pong(on|off)`: Controls automatic pong replies in each active session loop.
- Option `keepalive_interval(Seconds)`: Schedules empty ping messages when a peer stays silent for the given positive number of seconds.
- Option `idle_timeout(Seconds)`: Closes a session with `status(1001, idle_timeout)` after the given positive number of seconds without an inbound message.
- Option `max_payload_length(Bytes)`: Rejects inbound frames whose declared payload length is greater than `Bytes` before allocating payload storage. Oversized frames are treated as `1009` close errors in each session loop. Use a non-negative integer.

`request_shutdown/1`

Requests shutdown of a registry-backed server loop started with serve_until_shutdown/5-6 for the specified control term and wakes any blocked accept call so the loop can terminate portably.

Compilation flags:

`static`

Template:

`request_shutdown(Control)`

Mode and number of proofs:

`request_shutdown(+nonvar)` - `one_or_error`

Exceptions:

Registry-backed serving is not available on this backend:

`existence_error(http_websocket_server_service,registry)`

`Control` is a variable:

`instantiation_error`

`Control` is not registered for an open-ended WebSocket server loop:

`existence_error(http_websocket_service_shutdown_control,Control)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`shutdown_seed_/1`

Last allocated shutdown run identifier for open-ended WebSocket server loops.

Compilation flags:

`dynamic`

Template:

```
shutdown_seed_(RunId)
```

Mode and number of proofs:

```
shutdown_seed_(?positive_integer) - zero_or_one
```

---

shutdown_control_/3

Registered shutdown control terms and their associated listeners.

Compilation flags:

```
dynamic
```

Template:

```
shutdown_control_(Control,Listener,RunId)
```

Mode and number of proofs:

```
shutdown_control_(?nonvar,?compound,?positive_integer) - zero_or_more
```

---

shutdown_requested_/2

Recorded shutdown requests indexed by control term and run identifier.

Compilation flags:

```
dynamic
```

Template:

```
shutdown_requested_(Control,RunId)
```

Mode and number of proofs:

```
shutdown_requested_(?nonvar,?positive_integer) - zero_or_more
```

---

active_worker_/3

Active WebSocket session worker records indexed by control term and run identifier.

Compilation flags:

```
dynamic
```

Template:

```
active_worker_(Control,RunId,Worker)
```

Mode and number of proofs:

```
active_worker_(?nonvar,?positive_integer,?compound) - zero_or_more
```

Operators

(none)
object

1.146.5 `http_websocket_service(HTTPTransport,Role,TextRepresentation)`

- `HTTPTransport` - The object implementing `http_transport_protocol`.
- `Role` - Peer role for masking policy. Possible values are `client` and `server`.
- `TextRepresentation` - Text representation to be used for text messages and close reasons. Possible values are `atom` (default), `chars`, and `codes`.

Callback-driven WebSocket session loops over upgraded WebSocket connections provided by a selected `http_transport_protocol` implementation, including automatic close-handshake orchestration, optional auto-pong, keepalive, and idle-timeout policies.

Availability:
`logtalk_load(http_websocket_service(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls, threaded`

Extends:
`public` *http_websocket_session(Role,TextRepresentation)*
Uses:
http_websocket_service_loop_options
http_websocket_service_registry
list
os
user

Remarks:

- Option precedence: When the same loop option is given multiple times, the first occurrence is used.
- Timed policies: Keepalive and idle-timeout deadlines are tracked using wall-clock time, making the loop robust to scheduling jitter and slow reads.

Inherited public predicates:
initial_state/1 is_message/1 is_state/1 message/3 payload/2 read_message/2 read_message/4 read_message/5 read_message/6 type/2 write_message/2 write_message/3 write_message/4 write_message/5

- *Public predicates*
  - *run_session/3*
  - *run_session/4*
- *Protected predicates*
  - *run_session_connection/5*

- validate_non_negative_integer_option/4
- *Private predicates*
- *Operators*

Public predicates

run_session/3

Runs a higher-level session loop on an upgraded WebSocket connection produced by the selected transport parameterization until the close handshake completes or the peer closes the stream, then closes the upgraded connection automatically.

Compilation flags:

`static`

Template:

`run_session(Connection,Handler,State)`

Mode and number of proofs:

`run_session(+compound,+object_identifier,-compound)` - `one_or_error`

Exceptions:

`Connection` is a variable:

`instantiation_error`

`Connection` is not a valid upgraded WebSocket connection handle:

`domain_error(http_socket_transport_connection,Connection)`

`Handler` is a variable:

`instantiation_error`

`Handler` is not a valid WebSocket service handler:

`domain_error(http_websocket_service_handler,Handler)`

The session role parameter is neither `client` nor `server`:

`domain_error(http_websocket_session_role,_Role_)`

The inbound frame/message sequence violates RFC 6455 session rules:

`domain_error(http_websocket_session_sequence,Frame)`

The inbound frame violates the current peer masking policy:

`domain_error(http_websocket_session_masking,Frame)`

The inbound frame uses unsupported extensions:

`domain_error(http_websocket_session_extensions,Frame)`

The inbound frame is invalid:

`domain_error(http_websocket_frame,Frame)`

The inbound message text is invalid:

`domain_error(http_websocket_message_text,Payload)`

`Handler` returns a variable reply list:

`instantiation_error`

`Handler` returns a non-list reply list:

`type_error(list,Replies)`

`Handler` returns an invalid reply:

`domain_error(http_websocket_service_handler_reply,Reply)`

Remarks:

- Handler protocol: The handler object must conform to the `http_websocket_service_handler_protocol` protocol and return a list of normalized reply messages for each received message.
- Connection ownership: This predicate takes ownership of the upgraded connection and closes it before succeeding or rethrowing any exception.

`run_session/4`

Runs a higher-level session loop on an upgraded WebSocket connection produced by the selected transport parameterization using the given loop options, then closes the upgraded connection automatically.

Compilation flags:

`static`

Template:

`run_session(Connection,Handler,State,Options)`

Mode and number of proofs:

`run_session(+compound,+object_identifier,-compound,+list)` - `one_or_error`

Exceptions:

`Connection` is a variable:

`instantiation_error`

`Connection` is not a valid upgraded WebSocket connection handle:

`domain_error(http_socket_transport_connection,Connection)`

`Handler` is a variable:

`instantiation_error`

`Handler` is not a valid WebSocket service handler:

`domain_error(http_websocket_service_handler,Handler)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid WebSocket service loop option:

`domain_error(http_websocket_service_option,Option)`

Timed session-loop options are not available on this backend:

`existence_error(http_websocket_service,timing)`

The session role parameter is neither `client` nor `server`:

`domain_error(http_websocket_session_role,_Role_)`

The inbound frame/message sequence violates RFC 6455 session rules:

`domain_error(http_websocket_session_sequence,Frame)`

The inbound frame violates the current peer masking policy:

`domain_error(http_websocket_session_masking,Frame)`

The inbound frame uses unsupported extensions:

`domain_error(http_websocket_session_extensions,Frame)`

The inbound frame is invalid:

`domain_error(http_websocket_frame,Frame)`

The inbound message text is invalid:

`domain_error(http_websocket_message_text,Payload)`

`Handler` returns a variable reply list:

`instantiation_error`

`Handler` returns a non-list reply list:

`type_error(list,Replies)`

`Handler` returns an invalid reply:

`domain_error(http_websocket_service_handler_reply,Reply)`

Remarks:

- Option `auto_pong(on)`: Automatically writes pong replies while still forwarding ping messages to the handler.
- Option `auto_pong(off)`: Disables automatic pong replies. This is the default.
- Option `keepalive_interval(Seconds)`: When no inbound message is received for the given positive number of seconds, writes an empty ping message and continues the loop. This option requires backend thread support.
- Option `idle_timeout(Seconds)`: When no inbound message is received for the given positive number of seconds, writes `message(close, status(1001, idle_timeout))` and waits up to one additional idle-timeout interval for the peer to complete the close handshake before stopping. This option requires backend thread support.
- Option `max_payload_length(Bytes)`: Rejects inbound frames whose declared payload length is greater than `Bytes` before allocating payload storage. Oversized frames are treated as `1009` close errors in the session loop. Use a non-negative integer.

Protected predicates

`run_session_connection/5`

Protected helper that runs the callback-driven session loop on an upgraded WebSocket connection starting from the given session state and using the given loop options. Connection ownership is handled by the caller.

Compilation flags:

`static`

Template:

`run_session_connection(Connection,HandlerDescriptor,State,UpdatedState,Options)`

Mode and number of proofs:

`run_session_connection(+compound,+compound,+compound,-compound,+list)` - `one_or_error`

Exceptions:

`Connection` is a variable:

`instantiation_error`

`Connection` is not a valid upgraded WebSocket connection handle:

`domain_error(http_socket_transport_connection,Connection)`

`HandlerDescriptor` is not a valid WebSocket service handler descriptor:

`domain_error(http_websocket_service_handler,HandlerDescriptor)`

The handler in `HandlerDescriptor` is a variable:

`instantiation_error`

The handler in `HandlerDescriptor` is not a valid WebSocket service handler:

```
domain_error(http_websocket_service_handler,Handler)
```

`State` is a variable:

```
instantiation_error
```

`State` is not a valid WebSocket session state:

```
domain_error(http_websocket_session_state,State)
```

`Options` is a variable or a partial list:

```
instantiation_error
```

`Options` is neither a variable nor a list:

```
type_error(list,Options)
```

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

`Options` contains an invalid WebSocket service loop option:

```
domain_error(http_websocket_service_option,Option)
```

Timed session-loop options are not available on this backend:

```
existence_error(http_websocket_service,timing)
```

The session role parameter is neither `client` nor `server`:

```
domain_error(http_websocket_session_role,_Role_)
```

The inbound frame/message sequence violates RFC 6455 session rules:

```
domain_error(http_websocket_session_sequence,Frame)
```

The inbound frame violates the current peer masking policy:

```
domain_error(http_websocket_session_masking,Frame)
```

The inbound frame uses unsupported extensions:

```
domain_error(http_websocket_session_extensions,Frame)
```

The inbound frame is invalid:

```
domain_error(http_websocket_frame,Frame)
```

The inbound message text is invalid:

```
domain_error(http_websocket_message_text,Payload)
```

The handler returns a variable reply or action list:

```
instantiation_error
```

The handler returns a non-list reply list:

```
type_error(list,Replies)
```

The handler returns a non-list action list:

```
type_error(list,Actions)
```

The handler returns an invalid reply:

```
domain_error(http_websocket_service_handler_reply,Reply)
```

The handler returns an invalid registry action:

```
domain_error(http_websocket_service_handler_action,Action)
```

The registry in `HandlerDescriptor` is not an open WebSocket service registry handle:

```
domain_error(http_websocket_service_registry,Registry)
```

The session in `HandlerDescriptor` is not a valid WebSocket service registry session handle:

```
domain_error(http_websocket_service_registry_session,Session)
```

The session in `HandlerDescriptor` is not registered in the registry:

```
existence_error(http_websocket_service_registry_session,
websocket_session(RegistryId,SessionId))
```

validate_non_negative_integer_option/4

Validates a non-negative integer option value for the given domain and option name.

Compilation flags:
`static`

Template:
`validate_non_negative_integer_option(Domain,Name,Value,ValidatedValue)`
Mode and number of proofs:
`validate_non_negative_integer_option(+atom,+atom,@term,-integer)` - `one_or_error`

Exceptions:
`Value` is not a non-negative integer option value for the given domain and option name:
`domain_error(Domain,Option)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.146.6 http_websocket_service_handler_protocol

Protocol for callback objects used by the callback-driven WebSocket service loops.

Availability:
`logtalk_load(http_websocket_service(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *handle/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

handle/2

Processes a received normalized WebSocket message and returns a list of zero or more session actions. Plain normalized messages are written back on the same session before the next read. When used with the registry-backed server helper, the list may also contain the action wrappers `reply(Message)`, `broadcast(Message)`, and `broadcast_others(Message)`.

Compilation flags:

`static`

Template:

`handle(Message,Replies)`

Mode and number of proofs:

`handle(+compound,-list(compound))` - `one_or_error`

Exceptions:

`Message` is not a valid normalized WebSocket message term:

`domain_error(http_websocket_message,Message)`

`Replies` is a variable:

`instantiation_error`

`Replies` is neither a variable nor a list:

`type_error(list,Replies)`

An element `Reply` of the list `Replies` is not a valid reply message:

`domain_error(http_websocket_service_handler_reply,Reply)`

An element `Reply` of the list `Replies` is not a valid registry-backed action:

`domain_error(http_websocket_service_handler_action,Reply)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.146.7 `http_websocket_service_loop_options`

Auxiliary object defining the supported session-loop options and default values for callback-driven WebSocket services.

Availability:
`logtalk_load(http_websocket_service(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-12

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *merged_options/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`merged_options/2`

Merges validated session-loop options with the declared default options.

Compilation flags:
`static`

Template:
`merged_options(UserOptions,MergedOptions)`

Mode and number of proofs:
`merged_options(+list(compound),-list(compound))` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.146.8 `http_websocket_service_registry`

Session registry predicates for active WebSocket sessions and queued broadcast delivery.

Availability:
`logtalk_load(http_websocket_service(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static, context_switching_calls`

Uses:
http_websocket_messages
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *open/1*
  - *close/1*
  - *register/2*
  - *unregister/2*
  - *session/2*

    - *sessions/2*
    - *session_count/2*
    - *send/3*
    - *broadcast/2*
    - *broadcast_except/3*
    - *take_pending/3*
  - *Protected predicates*
  - *Private predicates*
    - *registry_seed_/1*
    - *registry_state_/2*
    - *registry_session_/3*
  - *Operators*

Public predicates

open/1

Opens a new empty WebSocket session registry.

Compilation flags:
`static`

Template:
`open(Registry)`
Mode and number of proofs:
`open(-compound)` - `one`

---

close/1

Closes a session registry and removes all registered sessions and queued messages.

Compilation flags:
`static`

Template:
`close(Registry)`
Mode and number of proofs:
`close(+compound)` - `one_or_error`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an open WebSocket service registry handle:
`domain_error(http_websocket_service_registry,Registry)`
`Registry` refers to a closed WebSocket service registry handle:

```
existence_error(http_websocket_service_registry,
session_registry(RegistryId))
```

---

register/2

Registers a new active session in the given registry and returns its session identifier.

Compilation flags:
`static`

Template:
`register(Registry,Session)`
Mode and number of proofs:
`register(+compound,-compound)` - `one_or_error`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an open WebSocket service registry handle:
`domain_error(http_websocket_service_registry,Registry)`
`Registry` refers to a closed WebSocket service registry handle:
`existence_error(http_websocket_service_registry,`
`session_registry(RegistryId))`

---

unregister/2

Removes a session from the given registry together with any queued outbound messages.

Compilation flags:
`static`

Template:
`unregister(Registry,Session)`
Mode and number of proofs:
`unregister(+compound,+compound)` - `one_or_error`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an open WebSocket service registry handle:
`domain_error(http_websocket_service_registry,Registry)`
`Registry` refers to a closed WebSocket service registry handle:
`existence_error(http_websocket_service_registry,`
`session_registry(RegistryId))`
`Session` is a variable or is not a session for `Registry`:

`instantiation_error`

`Session` is not a valid WebSocket service registry session handle:

`domain_error(http_websocket_service_registry_session,Session)`

`Session` is not registered in `Registry`:

`existence_error(http_websocket_service_registry_session, websocket_session(RegistryId,SessionId))`

---

session/2

Enumerates active session identifiers registered in the given registry.

Compilation flags:

`static`

Template:

`session(Registry,Session)`

Mode and number of proofs:

`session(+compound,?compound)` - `zero_or_more`

---

sessions/2

Returns the list of active session identifiers registered in the given registry.

Compilation flags:

`static`

Template:

`sessions(Registry,Sessions)`

Mode and number of proofs:

`sessions(+compound,-list(compound))` - `one_or_error`

Exceptions:

`Registry` is a variable:

`instantiation_error`

`Registry` is neither a variable nor an open WebSocket service registry handle:

`domain_error(http_websocket_service_registry,Registry)`

`Registry` refers to a closed WebSocket service registry handle:

`existence_error(http_websocket_service_registry, session_registry(RegistryId))`

---

session_count/2

Returns the number of active sessions currently registered in the given registry.

Compilation flags:

```
static
```

Template:

```
session_count(Registry,Count)
```

Mode and number of proofs:

```
session_count(+compound,-integer) - one_or_error
```

Exceptions:

`Registry` is a variable:

```
instantiation_error
```

`Registry` is neither a variable nor an open WebSocket service registry handle:

```
domain_error(http_websocket_service_registry,Registry)
```

`Registry` refers to a closed WebSocket service registry handle:

```
existence_error(http_websocket_service_registry,
session_registry(RegistryId))
```

---

send/3

Queues a normalized outbound message for the specified registered session.

Compilation flags:

```
static
```

Template:

```
send(Registry,Session,Message)
```

Mode and number of proofs:

```
send(+compound,+compound,+compound) - one_or_error
```

Exceptions:

`Registry` is not an open WebSocket service registry handle:

```
domain_error(http_websocket_service_registry,Registry)
```

`Session` is not a valid WebSocket service registry session handle:

```
domain_error(http_websocket_service_registry_session,Session)
```

`Session` is not registered in `Registry`:

```
existence_error(http_websocket_service_registry_session,
websocket_session(RegistryId,SessionId))
```

`Message` is not a valid normalized WebSocket message term:

```
domain_error(http_websocket_service_registry_message,Message)
```

---

broadcast/2

Queues a normalized outbound message for all currently registered sessions.

Compilation flags:
`static`

Template:
`broadcast(Registry,Message)`
Mode and number of proofs:
`broadcast(+compound,+compound)` - `one_or_error`

Exceptions:
`Registry` is not an open WebSocket service registry handle:
`domain_error(http_websocket_service_registry,Registry)`
`Message` is not a valid normalized WebSocket message term:
`domain_error(http_websocket_service_registry_message,Message)`

---

broadcast_except/3

Queues a normalized outbound message for all currently registered sessions except the specified session.

Compilation flags:
`static`

Template:
`broadcast_except(Registry,Session,Message)`
Mode and number of proofs:
`broadcast_except(+compound,+compound,+compound)` - `one_or_error`

Exceptions:
`Registry` is not an open WebSocket service registry handle:
`domain_error(http_websocket_service_registry,Registry)`
`Session` is not a valid WebSocket service registry session handle:
`domain_error(http_websocket_service_registry_session,Session)`
`Session` is not registered in `Registry`:
`existence_error(http_websocket_service_registry_session,`
`websocket_session(RegistryId,SessionId))`
`Message` is not a valid normalized WebSocket message term:
`domain_error(http_websocket_service_registry_message,Message)`

---

take_pending/3

Removes and returns the queued outbound messages for the specified registered session.

Compilation flags:

```
static
```

Template:

```
take_pending(Registry,Session,Messages)
```

Mode and number of proofs:

```
take_pending(+compound,+compound,-list(compound)) - one_or_error
```

Exceptions:

`Registry` is not an open WebSocket service registry handle:

```
domain_error(http_websocket_service_registry,Registry)
```

`Session` is not a valid WebSocket service registry session handle:

```
domain_error(http_websocket_service_registry_session,Session)
```

`Session` is not registered in `Registry`:

```
existence_error(http_websocket_service_registry_session,
websocket_session(RegistryId,SessionId))
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

registry_seed_/1

Last allocated registry identifier.

Compilation flags:

```
dynamic
```

Template:

```
registry_seed_(RegistryId)
```

Mode and number of proofs:

```
registry_seed_(?positive_integer) - zero_or_one
```

---

registry_state_/2

Per-registry next session identifier state.

Compilation flags:
```
dynamic
```

Template:
```
registry_state_(RegistryId,NextSessionId)
```
Mode and number of proofs:
```
registry_state_(?positive_integer,?non_negative_integer) - zero_or_more
```

---

registry_session_/3

Registered sessions and their queued outbound messages indexed by registry identifier. Messages are stored in reverse order to make enqueue operations constant time.

Compilation flags:
```
dynamic
```

Template:
```
registry_session_(RegistryId,SessionId,QueuedMessagesReversed)
```
Mode and number of proofs:
```
registry_session_(?positive_integer,?positive_integer,?list(compound)) -
zero_or_more
```

Operators

(none)

1.147 http_websocket_session

object

1.147.1 http_websocket_client_session

Stateful WebSocket session predicates for client-side use with atom text representation.

Availability:
```
logtalk_load(http_websocket_session(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:

`static, context_switching_calls`

Extends:
`public` *http_websocket_session(client,atom)*

Remarks:
(none)

Inherited public predicates:
initial_state/1 is_message/1 is_state/1 message/3 payload/2 read_message/2 read_message/4 read_message/5 read_message/6 type/2 write_message/2 write_message/3 write_message/4 write_message/5

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.147.2 `http_websocket_server_session`

Stateful WebSocket session predicates for server-side use with atom text representation.

Availability:
`logtalk_load(http_websocket_session(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static, context_switching_calls`

Extends:

`public` *http_websocket_session(server,atom)*

Remarks:
(none)

Inherited public predicates:
initial_state/1 is_message/1 is_state/1 message/3 payload/2 read_message/2 read_message/4 read_message/5 read_message/6 type/2 write_message/2 write_message/3 write_message/4 write_message/5

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.147.3 `http_websocket_session(Role,TextRepresentation)`

- `Role` - Peer role for masking policy. Possible values are `client` and `server`.
- `TextRepresentation` - Text representation to be used for text messages and close reasons. Possible values are `atom` (default), `chars`, and `codes`.

Stateful WebSocket session predicates that build on top of the message layer to surface interleaved control frames, apply role-aware masking policies, and track close-handshake state. The callback-driven session loops are provided by the http_websocket_service library.

Availability:
`logtalk_load(http_websocket_session(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static, context_switching_calls`

Extends:

`public` *http_websocket_messages(TextRepresentation)*

Uses:

crypto
http_websocket_frames
http_websocket_session_read_options
http_websocket_session_write_options
list

Remarks:

- Option precedence: When the same read or write option is given multiple times, the first occurrence is used.

Inherited public predicates:

is_message/1 *message/3* *payload/2* *read_message/2* *type/2*

- *Public predicates*
  - `initial_state/1`
  - `is_state/1`
  - `read_message/4`
  - `read_message/5`
  - `read_message/6`
  - `write_message/2`
  - `write_message/3`
  - `write_message/4`
  - `write_message/5`
- *Protected predicates*
  - `validate_role/0`
  - `read_session_message/5`
  - `apply_session_read/6`
  - `current_close_state/2`
  - `close_handshake_completed/1`
  - `best_effort_protocol_error_close/3`
  - `generate_masking_key/1`
- *Private predicates*
- *Operators*

Public predicates

`initial_state/1`

Returns the initial session state term to be threaded through stateful reads.

Compilation flags:

`static`

Template:

```
initial_state(State)
```

Mode and number of proofs:

```
initial_state(-compound) - one
```

---

is_state/1

True when the argument is a valid session state term.

Compilation flags:

```
static
```

Template:

```
is_state(State)
```

Mode and number of proofs:

```
is_state(@term) - zero_or_one
```

---

read_message/4

Reads the next session-level WebSocket message from a binary stream using the given input state. Fragmented text and binary messages are reassembled across calls, interleaved control frames are surfaced immediately, and close-message state transitions are tracked in the returned state.

Compilation flags:

```
static
```

Template:

```
read_message(Stream,State,UpdatedState,Message)
```

Mode and number of proofs:

```
read_message(+stream_or_alias,+compound,-compound,-term) - one_or_error
```

Exceptions:

`Stream` is a variable:

```
instantiation_error
```

The session role parameter is neither `client` nor `server`:

```
domain_error(http_websocket_session_role,_Role_)
```

`State` is not a valid WebSocket session state:

```
domain_error(http_websocket_session_state,State)
```

The inbound frame/message sequence violates RFC 6455 session rules:

```
domain_error(http_websocket_session_sequence,Frame)
```

The inbound frame violates the current peer masking policy:

```
domain_error(http_websocket_session_masking,Frame)
```

The inbound frame uses unsupported extensions:

```
domain_error(http_websocket_session_extensions,Frame)
```

The inbound frame is invalid:

```
domain_error(http_websocket_frame,Frame)
```

The inbound message text is invalid:

```
domain_error(http_websocket_message_text,Payload)
```

Remarks:

- End of file: Returns `end_of_file` only when the input state is idle and the stream is already exhausted.
- Masking policy: Incoming frame masking is validated against the session role: client sessions expect unmasked inbound frames and server sessions expect masked inbound frames.
- Peer close: When a close frame is read, any fragmented data message in progress is discarded. Subsequent reads from the resulting `close_received/1` or `closed/2` state are rejected.

---

read_message/5

Reads the next session-level WebSocket message from a binary input stream using the given state and automatically orchestrates the close handshake on the given binary output stream when a close message is received.

Compilation flags:

```
static
```

Template:

```
read_message(Input,Output,State,UpdatedState,Message)
```

Mode and number of proofs:

```
read_message(+stream_or_alias,+stream_or_alias,+compound,-compound,-term) -
one_or_error
```

Exceptions:

`Input` is a variable:

```
instantiation_error
```

The session role parameter is neither `client` nor `server`:

```
domain_error(http_websocket_session_role,_Role_)
```

`State` is not a valid WebSocket session state:

```
domain_error(http_websocket_session_state,State)
```

The inbound frame/message sequence violates RFC 6455 session rules:

```
domain_error(http_websocket_session_sequence,Frame)
```

The inbound frame violates the current peer masking policy:

```
domain_error(http_websocket_session_masking,Frame)
```

The inbound frame uses unsupported extensions:

```
domain_error(http_websocket_session_extensions,Frame)
```

The inbound frame is invalid:

```
domain_error(http_websocket_frame,Frame)
```

The inbound message text is invalid:

```
domain_error(http_websocket_message_text,Payload)
```

Remarks:

- Close handshake: When a close message is received and no close message has yet been sent for the tracked session state, a matching close message is written automatically on the output stream and the returned state records the completed handshake.

---

read_message/6

Reads the next session-level WebSocket message from a binary input stream using the given state, automatically orchestrates the close handshake on the given binary output stream, and applies optional automatic control-message policies.

Compilation flags:

`static`

Template:

`read_message(Input,Output,State,UpdatedState,Message,Options)`

Mode and number of proofs:

`read_message(+stream_or_alias,+stream_or_alias,+compound,-compound,-term,+list)` - `one_or_error`

Exceptions:

`Input` is a variable:

`instantiation_error`

The session role parameter is neither `client` nor `server`:

`domain_error(http_websocket_session_role,_Role_)`

`State` is not a valid WebSocket session state:

`domain_error(http_websocket_session_state,State)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid read option:

`domain_error(http_websocket_session_read_option,Option)`

The inbound frame/message sequence violates RFC 6455 session rules:

`domain_error(http_websocket_session_sequence,Frame)`

The inbound frame violates the current peer masking policy:

`domain_error(http_websocket_session_masking,Frame)`

The inbound frame uses unsupported extensions:

`domain_error(http_websocket_session_extensions,Frame)`

The inbound payload length exceeds the configured limit:

`domain_error(http_websocket_payload_length_limit,PayloadLength)`

Remarks:

- Option `auto_pong(on)`: Automatically writes a pong message with the same payload when a ping message is read while still returning the ping message to the caller.
- Option `auto_pong(off)`: Disables automatic pong replies. This is the default.
- Option `max_payload_length(Bytes)`: Rejects inbound frames whose declared payload length is greater than `Bytes` before allocating payload storage. Use a non-negative integer.

---

write_message/2

Stateless convenience wrapper that writes one validated WebSocket message using the masking policy implied by the session role. Client sessions mask all outgoing frames; server sessions leave them unmasked.

Compilation flags:

`static`

Template:

`write_message(Stream,Message)`

Mode and number of proofs:

`write_message(+stream_or_alias,+compound)` - `one_or_error`

Exceptions:

`Stream` is a variable:

`instantiation_error`

The session role parameter is neither `client` nor `server`:

`domain_error(http_websocket_session_role,_Role_)`

`Message` is not a valid normalized WebSocket message term:

`domain_error(http_websocket_message,Message)`

`Message` contains invalid text:

`domain_error(http_websocket_message_text,Text)`

The generated outbound frame is invalid:

`domain_error(http_websocket_frame,Frame)`

---

write_message/3

Stateless convenience wrapper that writes one validated WebSocket message using the masking policy implied by the session role and the given write options.

Compilation flags:

`static`

Template:

`write_message(Stream,Message,Options)`

Mode and number of proofs:

`write_message(+stream_or_alias,+compound,+list)` - `one_or_error`

Exceptions:

`Stream` is a variable:

`instantiation_error`

The session role parameter is neither `client` nor `server`:

`domain_error(http_websocket_session_role,_Role_)`

`Message` is not a valid normalized WebSocket message term:

`domain_error(http_websocket_message,Message)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid write option:
`domain_error(http_websocket_session_write_option,Option)`
The generated outbound frame is invalid:
`domain_error(http_websocket_frame,Frame)`

Remarks:

- Option `fragment_size(Size)`: When writing `text` or `binary` messages, split the payload into frames of at most `Size` bytes. Control messages always remain single final frames.

---

write_message/4

Writes one validated WebSocket message using the masking policy implied by the session role and updates the given session state with any close-handshake transition caused by the write.

Compilation flags:
`static`

Template:
`write_message(Stream,State,UpdatedState,Message)`
Mode and number of proofs:
`write_message(+stream_or_alias,+compound,-compound,+compound)` - `one_or_error`

Exceptions:
`Stream` is a variable:
`instantiation_error`
The session role parameter is neither `client` nor `server`:
`domain_error(http_websocket_session_role,_Role_)`
`State` is not a valid WebSocket session state:
`domain_error(http_websocket_session_state,State)`
`Message` is not a valid normalized WebSocket message term:
`domain_error(http_websocket_message,Message)`
The write violates the current close-handshake state:
`domain_error(http_websocket_session_state,State)`

Remarks:

- Closing state: After a peer close is recorded, only the matching close reply may be written. After a local close is recorded, further data messages are rejected.

---

write_message/5

Writes one validated WebSocket message using the masking policy implied by the session role and the given write options while updating the given session state with any close-handshake transition caused by the write.

Compilation flags:

`static`

Template:

`write_message(Stream,State,UpdatedState,Message,Options)`

Mode and number of proofs:

`write_message(+stream_or_alias,+compound,-compound,+compound,+list)` - `one_or_error`

Exceptions:

`Stream` is a variable:

`instantiation_error`

The session role parameter is neither `client` nor `server`:

`domain_error(http_websocket_session_role,_Role_)`

`State` is not a valid WebSocket session state:

`domain_error(http_websocket_session_state,State)`

`Message` is not a valid normalized WebSocket message term:

`domain_error(http_websocket_message,Message)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid write option:

`domain_error(http_websocket_session_write_option,Option)`

The write violates the current close-handshake state:

`domain_error(http_websocket_session_state,State)`

Remarks:

- Option `fragment_size(Size)`: When writing `text` or `binary` messages, split the payload into frames of at most `Size` bytes. Control messages always remain single final frames.

Protected predicates

validate_role/0

Validates the session role parameter. Throws a domain error when the role is neither `client` nor `server`.

Compilation flags:

`static`

Mode and number of proofs:

`validate_role` - `one_or_error`

Exceptions:

The session role parameter is neither `client` nor `server`:

```
domain_error(http_websocket_session_role,_Role_)
```

---

read_session_message/5

Protected helper that validates the session role and state and reads the next session-level message, returning the updated pending-fragment bookkeeping term and the message. Close-state transitions are left to the caller.

Compilation flags:

```
static
```

Template:

```
read_session_message(Stream,State,MaxPayloadLength,Pending,Message)
```

Mode and number of proofs:

```
read_session_message(+stream_or_alias,+compound,+term,-compound,-term) -
one_or_error
```

Exceptions:

The session role parameter is neither `client` nor `server`:

```
domain_error(http_websocket_session_role,_Role_)
```

`State` is not a valid WebSocket session state:

```
domain_error(http_websocket_session_state,State)
```

The inbound frame/message sequence violates RFC 6455 session rules:

```
domain_error(http_websocket_session_sequence,Frame)
```

The inbound frame violates the current peer masking policy:

```
domain_error(http_websocket_session_masking,Frame)
```

The inbound frame uses unsupported extensions:

```
domain_error(http_websocket_session_extensions,Frame)
```

---

apply_session_read/6

Protected helper that applies the close-state and automatic control-message transitions for a message read outside this object and returns the updated session state.

Compilation flags:

```
static
```

Template:

```
apply_session_read(Output,State,AutoPong,Pending,Message,UpdatedState)
```

Mode and number of proofs:

```
apply_session_read(+stream_or_alias,+compound,+atom,+compound,+term,-compound) -
one_or_error
```

Exceptions:

`State` is not a valid WebSocket session state:
`domain_error(http_websocket_session_state,State)`
`AutoPong` is not a supported automatic pong setting:
`domain_error(http_websocket_session_read_option,auto_pong(AutoPong))`
`Message` is not a valid normalized WebSocket message term:
`domain_error(http_websocket_message,Message)`

---

current_close_state/2

Protected helper that returns the close-handshake component of a session state term. Open states are reported as `open`.

Compilation flags:
`static`

Template:
`current_close_state(State,CloseState)`
Mode and number of proofs:
`current_close_state(+compound,-compound)` - `one`

---

close_handshake_completed/1

Protected helper that is true when the given session state records a completed close handshake.

Compilation flags:
`static`

Template:
`close_handshake_completed(State)`
Mode and number of proofs:
`close_handshake_completed(+compound)` - `zero_or_one`

---

best_effort_protocol_error_close/3

Protected helper that writes a best-effort close message mapping the given protocol error to its close status code when the session close state still allows it. Write errors are ignored.

Compilation flags:
`static`

Template:
`best_effort_protocol_error_close(Output,State,Error)`
Mode and number of proofs:
`best_effort_protocol_error_close(+stream_or_alias,+compound,+callable)` - `one`

---

generate_masking_key/1

Protected hook that returns the four-byte masking key to use for outgoing client frames. The default implementation delegates to `crypto::random_bytes/2`.

Compilation flags:
`static`

Template:
`generate_masking_key(Key)`
Mode and number of proofs:
`generate_masking_key(-list(byte))` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.147.4 http_websocket_session_read_options

Auxiliary object defining the supported read options and default values for HTTP WebSocket sessions.

Availability:
`logtalk_load(http_websocket_session(loader))`

Author: Paulo Moura
Version: 1:0:2
Date: 2026-08-12

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*

  - *merged_options/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

merged_options/2

Merges validated read options with the declared default options.

Compilation flags:
`static`

Template:
`merged_options(UserOptions,MergedOptions)`
Mode and number of proofs:
`merged_options(+list(compound),-list(compound))` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.147.5 http_websocket_session_write_options

Auxiliary object defining the supported write options and default values for HTTP WebSocket sessions.

Availability:
`logtalk_load(http_websocket_session(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-12

Compilation flags:
`static, context_switching_calls`

Imports:

`public` *options*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *merged_options/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

merged_options/2

Merges validated write options with the declared default options.

Compilation flags:
`static`

Template:
`merged_options(UserOptions,MergedOptions)`
Mode and number of proofs:
`merged_options(+list(compound),-list(compound))` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.148 ica_projection

object

1.148.1 `ica_projection`

Independent Component Analysis reducer for continuous datasets using a portable FastICA-style solver with symmetric eigendecomposition whitening.

Availability:

```
logtalk_load(ica_projection(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-05-23

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public dimension_reducer_common
```

Uses:

format
linear_algebra
list
numberlist
pairs
type

Remarks:

(none)

Inherited public predicates:

check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

nmf_projection, *pca_projection*, *random_projection*, *truncated_svd_projection*

1.149 ids

object

1.149.1 `ids`

Generator of random identifiers represented as atoms with 160 bits (20 bytes) of randomness.

Availability:
`logtalk_load(ids(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *ids(atom,20)*

Remarks:
(none)

Inherited public predicates:
generate/1

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ids(Representation,Bytes), *cuid2*, *ksuid*, *nanoid*, *snowflakeid*, *sqids*, *typeid*, *uuid*, *ulid*

object

1.149.2 `ids(Representation,Bytes)`

- `Representation` - Text representation for the identifier. Possible values are `atom`, `chars`, and `codes`.
- `Bytes` - Number of bytes of randomness.

Generator of random identifiers.

Availability:
`logtalk_load(ids(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Uses:
base64
crypto

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *generate/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

generate/1

Generate a random identifier.

Compilation flags:
`static`

Template:
`generate(Identifier)`

Mode and number of proofs:
`generate(--textids)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ids, *nanoid(Representation,Size,Alphabet)*, *cuid2(Representation,Size,Alphabet)*, *ksuid(Representation,Alphabet)*, *uuid(Representation)*, *snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node)*, *sqids(Representation)*, *typeid(Representation)*, *ulid(Representation)*

1.150 ieee_754

object

1.150.1 `ieee_754(Precision,ByteOrder,NaNRepresentation)`

- `Precision` - Floating-point format precision. Supported values are `half`, `single`, and `double`.
- `ByteOrder` - Byte order used when parsing and generating byte sequences. Supported values are `big` and `little`.
- `NaNRepresentation` - NaN decoding and generation policy. Supported values are `canonical` and `payloads`.

IEEE 754 floating-point encoder and decoder.

Availability:
`logtalk_load(ieee_754(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-08-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ieee_754_protocol*
Imports:
`public` *ieee_754_common(Precision,ByteOrder)*
Uses:
byte_order
type

Remarks:
(none)

Inherited public predicates:
byte_count/1 *exactly_representable/1* *generate/2* *generate/3* *nan_representation/1* *order/1* *parse/2* *precision/1* *valid/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.150.2 `ieee_754_common(Precision,ByteOrder)`

Shared IEEE 754 exact bit and byte handling predicates for the high-level codec and low-level field inspection objects.

Availability:
`logtalk_load(ieee_754(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-08-04

Compilation flags:
`static`

Uses:
byte_order
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *source_bits/2*
  - *bits_fields/5*
  - *bits_finite_binary_rational/4*
  - *bits_nan_payload/2*
  - *bits_nan_kind/2*
  - *canonical_nan_bits/1*

- *nan_bits/1*
- *valid_bits/1*
- *zero_bits/2*
- *zero_from_sign/2*
- *apply_float_sign/3*
- *precision_spec/5*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

source_bits/2

Validates and normalizes a source term into canonical-order unsigned integer bits for the selected precision and byte order.

Compilation flags:
`static`

Template:
`source_bits(Source,Bits)`

Mode and number of proofs:
`source_bits(++compound,--integer)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is not a variable but uses `bytes(Bytes)` or `bits(Bits)` with an encoding incompatible with the selected precision:
`domain_error(ieee_754_encoding,Source)`
`Source` is not a valid IEEE 754 source term:
`domain_error(ieee_754_source,Source)`

---

bits_fields/5

Extracts exact IEEE 754 sign, exponent bits, mantissa bits, and a class atom from a validated bit pattern.

Compilation flags:
`static`

Template:
`bits_fields(Bits,Sign,ExponentBits,MantissaBits,Class)`

Mode and number of proofs:

```
bits_fields(+integer,-integer,-integer,-integer,-atom) - one
```

---

bits_finite_binary_rational/4

Extracts an exact finite IEEE 754 value from a validated bit pattern as `(-1)^Sign * Significand * 2^Exponent`.

Compilation flags:

```
static
```

Template:

```
bits_finite_binary_rational(Bits,Sign,Significand,Exponent)
```

Mode and number of proofs:

```
bits_finite_binary_rational(+integer,-integer,-integer,-integer) - zero_or_one
```

---

bits_nan_payload/2

Extracts the raw NaN mantissa payload bits from a validated bit pattern.

Compilation flags:

```
static
```

Template:

```
bits_nan_payload(Bits,PayloadBits)
```

Mode and number of proofs:

```
bits_nan_payload(+integer,-integer) - zero_or_one
```

---

bits_nan_kind/2

Classifies a validated NaN bit pattern as `quiet` or `signaling` using the quiet/signaling discriminator bit for the selected precision.

Compilation flags:

```
static
```

Template:

```
bits_nan_kind(Bits,Kind)
```

Mode and number of proofs:

```
bits_nan_kind(+integer,-atom) - zero_or_one
```

---

canonical_nan_bits/1

Returns the canonical quiet NaN bit pattern for the selected precision.

Compilation flags:
`static`

Template:
`canonical_nan_bits(Bits)`
Mode and number of proofs:
`canonical_nan_bits(-integer)` - `one`

---

nan_bits/1

True when the validated bit pattern encodes a NaN value for the selected precision.

Compilation flags:
`static`

Template:
`nan_bits(Bits)`
Mode and number of proofs:
`nan_bits(+integer)` - `zero_or_one`

---

valid_bits/1

True when the integer fits in the selected precision bit width.

Compilation flags:
`static`

Template:
`valid_bits(Bits)`
Mode and number of proofs:
`valid_bits(+integer)` - `zero_or_one`

---

zero_bits/2

Returns the zero encoding bits with the requested sign bit.

Compilation flags:
`static`

Template:
`zero_bits(Sign,Bits)`
Mode and number of proofs:
`zero_bits(+integer,-integer)` - `one`

---

zero_from_sign/2

Constructs a positive or negative floating-point zero from a sign bit.

Compilation flags:
`static`

Template:
`zero_from_sign(Sign,Zero)`
Mode and number of proofs:
`zero_from_sign(+integer,-float)` - `one`

---

apply_float_sign/3

Applies a sign bit to a positive floating-point magnitude.

Compilation flags:
`static`

Template:
`apply_float_sign(Sign,Magnitude,Value)`
Mode and number of proofs:
`apply_float_sign(+integer,+float,-float)` - `one`

`precision_spec/5`

Returns the exponent width, mantissa width, exponent bias, and byte count for a supported IEEE 754 precision.

Compilation flags:
`static`

Template:
`precision_spec(Precision,ExponentWidth,MantissaWidth,Bias,ByteCount)`
Mode and number of proofs:
`precision_spec(+atom,-integer,-integer,-integer,-integer)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.150.3 `ieee_754_fields(Precision,ByteOrder)`

- `Precision` - Floating-point format precision. Supported values are `half`, `single`, and `double`.
- `ByteOrder` - Byte order used when reading byte sequences. Supported values are `big` and `little`.

IEEE 754 exact bit-field classifier and decomposition object.

Availability:
`logtalk_load(ieee_754(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-09

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ieee_754_fields_protocol*
Imports:
`public` *ieee_754_common(Precision,ByteOrder)*

Remarks:
(none)

Inherited public predicates:
byte_count/1 *classify/2* *fields/5* *finite_binary_rational/4* *nan_kind/2* *nan_payload/2* *order/1* *precision/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.150.4 `ieee_754_fields_protocol`

IEEE 754 exact bit-field support library protocol. Intended to be implemented by parameterized objects such as `ieee_754_fields(Precision, ByteOrder)`.

Availability:
`logtalk_load(ieee_754(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-09

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

      - *classify/2*
      - *fields/5*
      - *finite_binary_rational/4*
      - *nan_payload/2*
      - *nan_kind/2*
      - *precision/1*
      - *order/1*
      - *byte_count/1*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

classify/2

Classifies an IEEE 754 encoding source term as `zero`, `subnormal`, `normal`, `infinity`, or `not_a_number` for the selected precision and byte order. Supported source terms are `bytes(Bytes)` and `bits(Bits)`.

Compilation flags:
`static`

Template:
`classify(Source,Class)`
Mode and number of proofs:
`classify(++compound,--atom)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source term:
`domain_error(ieee_754_source,Source)`
`Source` does not contain a valid encoding for the selected object:
`domain_error(ieee_754_encoding,Source)`

---

fields/5

Extracts the exact IEEE 754 sign bit, exponent bits, mantissa bits, and classification from a source term.

Compilation flags:
`static`

Template:
`fields(Source,Sign,ExponentBits,MantissaBits,Class)`
Mode and number of proofs:
`fields(++compound,--integer,--integer,--integer,--atom)` - `one_or_error`

Exceptions:

`Source` is a variable:

```
instantiation_error
```

`Source` is neither a variable nor a valid source term:

```
domain_error(ieee_754_source,Source)
```

`Source` does not contain a valid encoding for the selected object:

```
domain_error(ieee_754_encoding,Source)
```

---

finite_binary_rational/4

Extracts the exact finite IEEE 754 value as `(-1)^Sign * Significand * 2^Exponent` from a source term. Zero encodings return a zero significand and exponent `0`.

Compilation flags:

```
static
```

Template:

```
finite_binary_rational(Source,Sign,Significand,Exponent)
```

Mode and number of proofs:

```
finite_binary_rational(++compound,--integer,--integer,--integer) - one_or_error
```

Exceptions:

`Source` is a variable:

```
instantiation_error
```

`Source` is neither a variable nor a valid source term:

```
domain_error(ieee_754_source,Source)
```

`Source` does not contain a valid encoding for the selected object:

```
domain_error(ieee_754_encoding,Source)
```

`Source` does not encode a finite IEEE 754 value:

```
domain_error(ieee_754_finite_encoding,Source)
```

---

nan_payload/2

Extracts the raw NaN mantissa payload bits from a source term.

Compilation flags:

```
static
```

Template:

```
nan_payload(Source,PayloadBits)
```

Mode and number of proofs:

```
nan_payload(++compound,--integer) - one_or_error
```

Exceptions:

`Source` is a variable:

instantiation_error
Source is neither a variable nor a valid source term:
domain_error(ieee_754_source,Source)
Source does not contain a valid encoding for the selected object:
domain_error(ieee_754_encoding,Source)
Source does not encode a NaN value:
domain_error(ieee_754_nan_encoding,Source)

---

nan_kind/2

Classifies a NaN source term as quiet or signaling using the exact IEEE 754 quiet/signaling discriminator bit for the selected precision.

Compilation flags:
static

Template:
nan_kind(Source,Kind)
Mode and number of proofs:
nan_kind(++compound,--atom) - one_or_error

Exceptions:
Source is a variable:
instantiation_error
Source is neither a variable nor a valid source term:
domain_error(ieee_754_source,Source)
Source does not contain a valid encoding for the selected object:
domain_error(ieee_754_encoding,Source)
Source does not encode a NaN value:
domain_error(ieee_754_nan_encoding,Source)

---

precision/1

Returns the selected IEEE 754 precision. Expected values are half, single, or double.

Compilation flags:
static

Template:
precision(Precision)
Mode and number of proofs:
precision(-atom) - one

---

order/1

Returns the selected byte order. Expected values are `big` or `little`.

Compilation flags:
`static`

Template:
`order(ByteOrder)`
Mode and number of proofs:
`order(-atom)` - `one`

---

byte_count/1

Returns the number of bytes used by the selected precision.

Compilation flags:
`static`

Template:
`byte_count(ByteCount)`
Mode and number of proofs:
`byte_count(-integer)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.150.5 ieee_754_protocol

IEEE 754 floating-point support library protocol. Intended to be implemented by parameterized objects such as `ieee_754(Precision, ByteOrder, NaNRepresentation)`.

Availability:
`logtalk_load(ieee_754(loader))`

Author: Paulo Moura

Version: 0:1:0
Date: 2026-05-09

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
  - *generate/3*
  - *valid/1*
  - *exactly_representable/1*
  - *precision/1*
  - *order/1*
  - *nan_representation/1*
  - *byte_count/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses an IEEE 754 value from a source term for the selected precision, byte order, and NaN representation policy. Supported source terms are `bytes(Bytes)` and `bits(Bits)`.

Compilation flags:
`static`

Template:
`parse(Source,Value)`
Mode and number of proofs:
`parse(++compound,--term)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source term:

`domain_error(ieee_754_source,Source)`
`Source` does not contain a valid encoding for the selected object:
`domain_error(ieee_754_encoding,Source)`

---

generate/2

Generates an IEEE 754 encoding for a value term using the selected precision and byte order. Supported sink terms are `bytes(Bytes)` and `bits(Bits)`. Finite values are encoded using the selected precision rounding semantics.

Compilation flags:
`static`

Template:
`generate(Sink,Value)`
Mode and number of proofs:
`generate(+compound,++term)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid sink term:
`domain_error(ieee_754_sink,Sink)`
`Value` is neither a valid finite float nor a supported IEEE 754 special value term:
`domain_error(ieee_754_value,Value)`
`Value` cannot be represented in the selected precision:
`domain_error(ieee_754_representation,Value)`

---

generate/3

Generates an IEEE 754 encoding as a list of bytes with an open tail. Intended for use by libraries that build larger binary payloads using difference lists.

Compilation flags:
`static`

Template:
`generate(Value,Bytes,Tail)`
Mode and number of proofs:
`generate(+term,-list(byte),--variable)` - `one_or_error`

Exceptions:
`Value` is neither a valid finite float nor a supported IEEE 754 special value term:
`domain_error(ieee_754_value,Value)`
`Value` cannot be represented in the selected precision:
`domain_error(ieee_754_representation,Value)`

---

valid/1

True iff the argument is a valid value term for the selected object. Valid terms are finite floats plus the special values supported by the selected NaN representation policy.

Compilation flags:
`static`

Template:
`valid(Value)`
Mode and number of proofs:
`valid(@term)` - `zero_or_one`

---

exactly_representable/1

True iff the argument can be encoded in the selected precision and decoded again without loss of value information under the selected NaN representation policy. Intended for callers such as MessagePack that choose the smallest exact binary format.

Compilation flags:
`static`

Template:
`exactly_representable(Value)`
Mode and number of proofs:
`exactly_representable(@term)` - `zero_or_one`

---

precision/1

Returns the selected IEEE 754 precision. Expected values are `half`, `single`, or `double`.

Compilation flags:
`static`

Template:
`precision(Precision)`
Mode and number of proofs:
`precision(-atom)` - `one`

---

order/1

Returns the selected byte order. Expected values are `big` or `little`.

Compilation flags:
`static`

Template:
`order(ByteOrder)`
Mode and number of proofs:
`order(-atom)` - `one`

---

nan_representation/1

Returns the selected NaN representation policy. Expected values are `canonical` and `payloads`.

Compilation flags:
`static`

Template:
`nan_representation(NaNRepresentation)`
Mode and number of proofs:
`nan_representation(-atom)` - `one`

---

byte_count/1

Returns the number of bytes used by the selected precision.

Compilation flags:
`static`

Template:
`byte_count(ByteCount)`
Mode and number of proofs:
`byte_count(-integer)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.151 integer_partitions

object

1.151.1 `integer_partitions`

Implementation of integer partition operations.

Availability:
`logtalk_load(integer_partitions(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *integer_partitions_protocol*
Uses:
fast_random(Algorithm)
integer
list
natural

Remarks:
(none)

Inherited public predicates:
count_distinct_partitions/2 *count_distinct_partitions/3* *count_partitions/2* *count_partitions/3* *distinct_partition/2* *distinct_partition/3* *distinct_partition/4* *distinct_partition_index/3* *distinct_partition_index/4* *distinct_partition_index/5* *distinct_partitions/2* *distinct_partitions/3* *distinct_partitions/4* *next_distinct_partition/3* *next_distinct_partition/4* *next_partition/3* *next_partition/4* *nth_distinct_partition/3* *nth_distinct_partition/4* *nth_distinct_partition/5* *nth_partition/3* *nth_partition/4* *nth_partition/5* *partition/2* *partition/3* *partition/4* *partition_index/3* *partition_index/4* *partition_index/5* *partitions/2* *partitions/3* *partitions/4* *previous_distinct_partition/3* *previous_distinct_partition/4* *previous_partition/3* *previous_partition/4* *random_distinct_partition/2* *random_distinct_partition/3* *random_partition/2* *random_partition/3* *sample_distinct_partitions/3* *sample_distinct_partitions/4* *sample_partitions/3* *sample_partitions/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.151.2 `integer_partitions_protocol`

Protocol for integer partition operations.

Availability:
`logtalk_load(integer_partitions(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-27

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

    - *partitions/2*
    - *partition/2*
    - *partitions/3*
    - *partition/3*
    - *partitions/4*
    - *partition/4*
    - *distinct_partitions/2*
    - *distinct_partition/2*
    - *distinct_partitions/3*
    - *distinct_partition/3*
    - *distinct_partitions/4*
    - *distinct_partition/4*
    - *count_partitions/2*
    - *count_partitions/3*
    - *count_distinct_partitions/2*
    - *count_distinct_partitions/3*
    - *nth_partition/3*
    - *nth_partition/4*
    - *nth_partition/5*
    - *partition_index/3*
    - *partition_index/4*
    - *partition_index/5*
    - *nth_distinct_partition/3*
    - *nth_distinct_partition/4*
    - *nth_distinct_partition/5*
    - *distinct_partition_index/3*
    - *distinct_partition_index/4*
    - *distinct_partition_index/5*
    - *random_partition/2*
    - *random_partition/3*
    - *sample_partitions/3*
    - *sample_partitions/4*
    - *random_distinct_partition/2*
    - *random_distinct_partition/3*
    - *sample_distinct_partitions/3*
    - *sample_distinct_partitions/4*
    - *next_partition/3*
    - *next_partition/4*
    - *previous_partition/3*
    - *previous_partition/4*
    - *next_distinct_partition/3*
    - *next_distinct_partition/4*
    - *previous_distinct_partition/3*
    - *previous_distinct_partition/4*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

partitions/2

Generates all partitions of a non-negative integer using default order. Partitions are represented as non-increasing lists of positive integers that sum to the given integer. Zero has one partition, represented by the empty list of parts.

Compilation flags:
`static`

Template:
`partitions(N,Partitions)`
Mode and number of proofs:
`partitions(+non_negative_integer,-list)` - `one`

---

partition/2

True iff the second argument is a partition of a non-negative integer using default order.

Compilation flags:
`static`

Template:
`partition(N,Partition)`
Mode and number of proofs:
`partition(+non_negative_integer,-list)` - `one_or_more`

---

partitions/3

Generates either all partitions of an integer with the given order (when the second argument is `default` or `lexicographic`) or all partitions with an exact number of parts using default order (when the second argument is an integer). Note that, unlike the `partitions` library (where the size constraint can occupy the first argument position, distinguishing it from a list by type), here the integer being partitioned and the exact part count are both integers, so the part count is always given as the second argument, distinguished from the order by type.

Compilation flags:
`static`

Template:
`partitions(N,KOrOrder,Partitions)`
Mode and number of proofs:
`partitions(+non_negative_integer,+atom,-list)` - `one`
`partitions(+non_negative_integer,+non_negative_integer,-list)` - `one`

---

partition/3

True iff the third argument is a partition generated either in the given order or with the given exact number of parts using default order.

Compilation flags:
`static`

Template:
`partition(N,KOrOrder,Partition)`
Mode and number of proofs:
`partition(+non_negative_integer,+atom,-list)` - `one_or_more`
`partition(+non_negative_integer,+non_negative_integer,-list)` - `one_or_more`

---

partitions/4

Generates all partitions of an integer with an exact number of parts and the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`partitions(N,K,Order,Partitions)`
Mode and number of proofs:
`partitions(+non_negative_integer,+non_negative_integer,+atom,-list)` - `one`

---

partition/4

True iff the fourth argument is a partition with an exact number of parts and the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`partition(N,K,Order,Partition)`
Mode and number of proofs:
`partition(+non_negative_integer,+non_negative_integer,+atom,-list)` - `one_or_more`

---

distinct_partitions/2

Generates all partitions of a non-negative integer into distinct (non-repeating) parts using default order. Unlike the `partitions` library, where "distinct" deduplicates equal-valued results from a position-sensitive generator, an integer partition has no position-sensitive input to begin with; here "distinct" is the standard integer partition notion of parts that are pairwise different (also known as strict partitions).

Compilation flags:

```
static
```

Template:

```
distinct_partitions(N,Partitions)
```

Mode and number of proofs:

```
distinct_partitions(+non_negative_integer,-list) - one
```

---

distinct_partition/2

True iff the second argument is a partition of a non-negative integer into distinct parts using default order.

Compilation flags:

```
static
```

Template:

```
distinct_partition(N,Partition)
```

Mode and number of proofs:

```
distinct_partition(+non_negative_integer,-list) - one_or_more
```

---

distinct_partitions/3

Generates either all partitions into distinct parts with the given order (when the second argument is `default` or `lexicographic`) or all partitions into an exact number of distinct parts using default order (when the second argument is an integer).

Compilation flags:

```
static
```

Template:

```
distinct_partitions(N,KOrOrder,Partitions)
```

Mode and number of proofs:

```
distinct_partitions(+non_negative_integer,+atom,-list) - one
distinct_partitions(+non_negative_integer,+non_negative_integer,-list) - one
```

---

distinct_partition/3

True iff the third argument is a partition into distinct parts generated either in the given order or with the given exact number of parts using default order.

Compilation flags:
`static`

Template:
`distinct_partition(N,KOrOrder,Partition)`
Mode and number of proofs:
`distinct_partition(+non_negative_integer,+atom,-list)` - `one_or_more`
`distinct_partition(+non_negative_integer,+non_negative_integer,-list)` - `one_or_more`

---

distinct_partitions/4

Generates all partitions of an integer into an exact number of distinct parts and the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_partitions(N,K,Order,Partitions)`
Mode and number of proofs:
`distinct_partitions(+non_negative_integer,+non_negative_integer,+atom,-list)` - `one`

---

distinct_partition/4

True iff the fourth argument is a partition into an exact number of distinct parts and the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_partition(N,K,Order,Partition)`
Mode and number of proofs:
`distinct_partition(+non_negative_integer,+non_negative_integer,+atom,-list)` - `one_or_more`

---

count_partitions/2

Counts the number of partitions of a non-negative integer.

Compilation flags:
`static`

Template:
`count_partitions(N,Count)`
Mode and number of proofs:
`count_partitions(+non_negative_integer,-non_negative_integer)` - `one`

---

count_partitions/3

Counts the number of partitions of a non-negative integer with an exact number of parts.

Compilation flags:
`static`

Template:
`count_partitions(N,K,Count)`
Mode and number of proofs:
`count_partitions(+non_negative_integer,+non_negative_integer,`
`-non_negative_integer)` - `zero_or_one`

---

count_distinct_partitions/2

Counts the number of partitions of a non-negative integer into distinct parts.

Compilation flags:
`static`

Template:
`count_distinct_partitions(N,Count)`
Mode and number of proofs:
`count_distinct_partitions(+non_negative_integer,-non_negative_integer)` - `one`

---

count_distinct_partitions/3

Counts the number of partitions of a non-negative integer into an exact number of distinct parts.

Compilation flags:
`static`

Template:
`count_distinct_partitions(N,K,Count)`
Mode and number of proofs:
`count_distinct_partitions(+non_negative_integer,+non_negative_integer,-non_negative_integer)` - `zero_or_one`

---

nth_partition/3

Returns the partition at a given zero-based index using default order.

Compilation flags:
`static`

Template:
`nth_partition(N,Index,Partition)`
Mode and number of proofs:
`nth_partition(+non_negative_integer,+integer,-list)` - `zero_or_one`

---

nth_partition/4

Returns either the partition at a given zero-based index in the given order (when the second argument is `default` or `lexicographic`) or the partition with an exact number of parts at a given zero-based index using default order (when the second argument is an integer).

Compilation flags:
`static`

Template:
`nth_partition(N,KOrOrder,Index,Partition)`
Mode and number of proofs:
`nth_partition(+non_negative_integer,+atom,+integer,-list)` - `zero_or_one`
`nth_partition(+non_negative_integer,+non_negative_integer,+integer,-list)` - `zero_or_one`

---

nth_partition/5

Returns the partition with an exact number of parts at a given zero-based index in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`nth_partition(N,K,Order,Index,Partition)`
Mode and number of proofs:
`nth_partition(+non_negative_integer,+non_negative_integer,+atom,+integer,-list)` - `zero_or_one`

---

partition_index/3

Returns the zero-based index of a partition using default order.

Compilation flags:
`static`

Template:
`partition_index(N,Partition,Index)`
Mode and number of proofs:
`partition_index(+non_negative_integer,+list,-integer)` - `zero_or_one`

---

partition_index/4

Returns either the zero-based index of a partition in the given order (when the second argument is `default` or `lexicographic`) or the zero-based index of a partition with an exact number of parts using default order (when the second argument is an integer).

Compilation flags:
`static`

Template:
`partition_index(N,KOrOrder,Partition,Index)`
Mode and number of proofs:
`partition_index(+non_negative_integer,+atom,+list,-integer)` - `zero_or_one`
`partition_index(+non_negative_integer,+non_negative_integer,+list,-integer)` - `zero_or_one`

---

partition_index/5

Returns the zero-based index of a partition with an exact number of parts in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`partition_index(N,K,Order,Partition,Index)`
Mode and number of proofs:
`partition_index(+non_negative_integer,+non_negative_integer,+atom,+list,-integer)` - `zero_or_one`

---

nth_distinct_partition/3

Returns the partition into distinct parts at a given zero-based index using default order.

Compilation flags:
`static`

Template:
`nth_distinct_partition(N,Index,Partition)`
Mode and number of proofs:
`nth_distinct_partition(+non_negative_integer,+integer,-list)` - `zero_or_one`

---

nth_distinct_partition/4

Returns either the partition into distinct parts at a given zero-based index in the given order (when the second argument is `default` or `lexicographic`) or the partition with an exact number of distinct parts at a given zero-based index using default order (when the second argument is an integer).

Compilation flags:
`static`

Template:
`nth_distinct_partition(N,KOrOrder,Index,Partition)`
Mode and number of proofs:
`nth_distinct_partition(+non_negative_integer,+atom,+integer,-list)` - `zero_or_one`
`nth_distinct_partition(+non_negative_integer,+non_negative_integer,+integer,-list)` - `zero_or_one`

---

nth_distinct_partition/5

Returns the partition with an exact number of distinct parts at a given zero-based index in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`nth_distinct_partition(N,K,Order,Index,Partition)`

Mode and number of proofs:
`nth_distinct_partition(+non_negative_integer,+non_negative_integer,+atom,+integer,-list)` - `zero_or_one`

---

distinct_partition_index/3

Returns the zero-based index of a partition into distinct parts using default order.

Compilation flags:
`static`

Template:
`distinct_partition_index(N,Partition,Index)`

Mode and number of proofs:
`distinct_partition_index(+non_negative_integer,+list,-integer)` - `zero_or_one`

---

distinct_partition_index/4

Returns either the zero-based index of a partition into distinct parts in the given order (when the second argument is `default` or `lexicographic`) or the zero-based index of a partition with an exact number of distinct parts using default order (when the second argument is an integer).

Compilation flags:
`static`

Template:
`distinct_partition_index(N,KOrOrder,Partition,Index)`

Mode and number of proofs:
`distinct_partition_index(+non_negative_integer,+atom,+list,-integer)` - `zero_or_one`
`distinct_partition_index(+non_negative_integer,+non_negative_integer,+list,-integer)` - `zero_or_one`

---

distinct_partition_index/5

Returns the zero-based index of a partition with an exact number of distinct parts in the given order: `default` or `lexicographic`.

Compilation flags:

`static`

Template:

`distinct_partition_index(N,K,Order,Partition,Index)`

Mode and number of proofs:

`distinct_partition_index(+non_negative_integer,+non_negative_integer,+atom,+list,-integer)` - `zero_or_one`

---

random_partition/2

Returns a random partition of a non-negative integer.

Compilation flags:

`static`

Template:

`random_partition(N,Partition)`

Mode and number of proofs:

`random_partition(+non_negative_integer,-list)` - `one`

---

random_partition/3

Returns a random partition of a non-negative integer with an exact number of parts.

Compilation flags:

`static`

Template:

`random_partition(N,K,Partition)`

Mode and number of proofs:

`random_partition(+non_negative_integer,+non_negative_integer,-list)` - `zero_or_one`

---

sample_partitions/3

Returns SampleCount random partitions of a non-negative integer, sampled with replacement.

Compilation flags:
```
static
```

Template:
```
sample_partitions(N,SampleCount,Samples)
```
Mode and number of proofs:
```
sample_partitions(+non_negative_integer,+non_negative_integer,-list) - zero_or_one
```

---

sample_partitions/4

Returns SampleCount random partitions of a non-negative integer with an exact number of parts, sampled with replacement.

Compilation flags:
```
static
```

Template:
```
sample_partitions(N,K,SampleCount,Samples)
```
Mode and number of proofs:
```
sample_partitions(+non_negative_integer,+non_negative_integer,
+non_negative_integer,-list) - zero_or_one
```

---

random_distinct_partition/2

Returns a random partition of a non-negative integer into distinct parts.

Compilation flags:
```
static
```

Template:
```
random_distinct_partition(N,Partition)
```
Mode and number of proofs:
```
random_distinct_partition(+non_negative_integer,-list) - one
```

---

random_distinct_partition/3

Returns a random partition of a non-negative integer with an exact number of distinct parts.

Compilation flags:
`static`

Template:
`random_distinct_partition(N,K,Partition)`
Mode and number of proofs:
`random_distinct_partition(+non_negative_integer,+non_negative_integer,-list)` - `zero_or_one`

---

sample_distinct_partitions/3

Returns SampleCount random partitions of a non-negative integer into distinct parts, sampled with replacement.

Compilation flags:
`static`

Template:
`sample_distinct_partitions(N,SampleCount,Samples)`
Mode and number of proofs:
`sample_distinct_partitions(+non_negative_integer,+non_negative_integer,-list)` - `zero_or_one`

---

sample_distinct_partitions/4

Returns SampleCount random partitions of a non-negative integer with an exact number of distinct parts, sampled with replacement.

Compilation flags:
`static`

Template:
`sample_distinct_partitions(N,K,SampleCount,Samples)`
Mode and number of proofs:
`sample_distinct_partitions(+non_negative_integer,+non_negative_integer,+non_negative_integer,-list)` - `zero_or_one`

---

next_partition/3

Returns the next partition value of a non-negative integer in lexicographic order.

Compilation flags:
`static`

Template:
`next_partition(N,Partition,Next)`
Mode and number of proofs:
`next_partition(+non_negative_integer,+list,-list)` - `zero_or_one`

---

next_partition/4

Returns the next partition value of a non-negative integer with an exact number of parts in lexicographic order.

Compilation flags:
`static`

Template:
`next_partition(N,K,Partition,Next)`
Mode and number of proofs:
`next_partition(+non_negative_integer,+non_negative_integer,+list,-list)` - `zero_or_one`

---

previous_partition/3

Returns the previous partition value of a non-negative integer in lexicographic order.

Compilation flags:
`static`

Template:
`previous_partition(N,Partition,Previous)`
Mode and number of proofs:
`previous_partition(+non_negative_integer,+list,-list)` - `zero_or_one`

---

previous_partition/4

Returns the previous partition value of a non-negative integer with an exact number of parts in lexicographic order.

Compilation flags:

```
static
```

Template:

```
previous_partition(N,K,Partition,Previous)
```

Mode and number of proofs:

```
previous_partition(+non_negative_integer,+non_negative_integer,+list,-list) -
zero_or_one
```

---

next_distinct_partition/3

Returns the next partition-into-distinct-parts value of a non-negative integer in lexicographic order.

Compilation flags:

```
static
```

Template:

```
next_distinct_partition(N,Partition,Next)
```

Mode and number of proofs:

```
next_distinct_partition(+non_negative_integer,+list,-list) - zero_or_one
```

---

next_distinct_partition/4

Returns the next partition-into-distinct-parts value of a non-negative integer with an exact number of parts in lexicographic order.

Compilation flags:

```
static
```

Template:

```
next_distinct_partition(N,K,Partition,Next)
```

Mode and number of proofs:

```
next_distinct_partition(+non_negative_integer,+non_negative_integer,+list,-list) -
zero_or_one
```

---

previous_distinct_partition/3

Returns the previous partition-into-distinct-parts value of a non-negative integer in lexicographic order.

Compilation flags:

```
static
```

Template:

```
previous_distinct_partition(N,Partition,Previous)
```

Mode and number of proofs:

```
previous_distinct_partition(+non_negative_integer,+list,-list) - zero_or_one
```

---

previous_distinct_partition/4

Returns the previous partition-into-distinct-parts value of a non-negative integer with an exact number of parts in lexicographic order.

Compilation flags:

```
static
```

Template:

```
previous_distinct_partition(N,K,Partition,Previous)
```

Mode and number of proofs:

```
previous_distinct_partition(+non_negative_integer,+non_negative_integer,+list,
-list) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.152 intermittent_demand_forecasting

object

1.152.1 intermittent_demand_forecasting

Croston, SBA, and TSB intermittent-demand forecasters with causal initialization, configurable missing observations, and immutable online updates.

Availability:
`logtalk_load(intermittent_demand_forecasting(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-14

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *forecaster_common*

Uses:
format
list
type

Remarks:
(none)

Inherited public predicates:
check_forecaster/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *forecast/3* *forecaster_options/2* *learn/2* *learn/3* *option/2* *option/3* *print_forecaster/1* *valid_forecaster/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *fitted_values/2*
  - *fitted_values/3*
  - *update/3*
- *Protected predicates*
- *Private predicates*
  - *replay_series/5*
  - *prepare_learning/4*
  - *normalize_coefficient_grid/2*
  - *select_methods/7*
  - *select_alphas/7*
  - *select_betas/7*
  - *fit_paired_series/4*
  - *learn_pair/4*
  - *paired_predictions/4*
  - *better_candidate/3*
  - *candidate_key/2*
  - *method_priority/2*

- *fit_series/4*
- *fit_series_output/5*
- *fitted_model/3*
- *learn_state/6*
- *score_observation/7*
- *demand_summary/3*
- *prepare_parameters/4*
- *positive_smoothing/4*
- *checked_forecaster/2*
- *valid_diagnostics/4*
- *extract_diagnostics/2*
- *rewrite_diagnostics/4*
- *diagnostic_position/2*
- *valid_error_metrics/5*

• *Operators*

Public predicates

`fitted_values/2`

Returns aligned one-step fitted values for `Dataset` using default options.

Compilation flags:

`static`

Template:

`fitted_values(Dataset,Values)`

Mode and number of proofs:

`fitted_values(+object_identifier,-list)` - `one_or_error`

Exceptions:

An observation index or the declared series length is a variable:

`instantiation_error`

The declared series length is not an integer:

`type_error(integer,DeclaredLength)`

The declared series length is not positive:

`domain_error(positive_integer,DeclaredLength)`

The dataset contains no observations:

`domain_error(non_empty_series,Dataset)`

The indices are not a gap-free 1-based sequence:

`domain_error(series_index_sequence,Dataset)`

The declared and observed lengths differ:

`consistency_error(series_length,DeclaredLength,ObservedLength)`

An observation is neither a variable nor a number:

`type_error(number,Observation)`

An observation is negative:

`domain_error(non_negative_number,Observation)`

All observations are missing:

`domain_error(insufficient_observations,Dataset)`

Fitting, error-total, or replay arithmetic raises an evaluation error:

`evaluation_error(Error)`

`fitted_values/3`

Returns aligned one-step fitted values for `Dataset` using the specified `Options`.

Compilation flags:

`static`

Template:

`fitted_values(Dataset,Values,Options)`

Mode and number of proofs:

`fitted_values(+object_identifier,-list,+list(compound))` - `one_or_error`

Exceptions:

An observation index or the declared series length is a variable:

`instantiation_error`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The declared series length is not an integer:

`type_error(integer,DeclaredLength)`

The declared series length is not positive:

`domain_error(positive_integer,DeclaredLength)`

The dataset contains no observations:

`domain_error(non_empty_series,Dataset)`

The indices are not a gap-free 1-based sequence:

`domain_error(series_index_sequence,Dataset)`

The declared and observed lengths differ:

`consistency_error(series_length,DeclaredLength,ObservedLength)`

An observation is neither a variable nor a number:

`type_error(number,Observation)`

An observation is negative:

`domain_error(non_negative_number,Observation)`

All observations are missing:

`domain_error(insufficient_observations,Dataset)`

Fitting, error-total, or replay arithmetic raises an evaluation error:

`evaluation_error(Error)`

update/3

Returns a new forecaster incorporating `Observation` without modifying `Forecaster`.

Compilation flags:
`static`

Template:
`update(Forecaster,Observation,UpdatedForecaster)`
Mode and number of proofs:
`update(+compound,@term,-compound)` - `one_or_error`

Exceptions:
`Forecaster` is a variable:
`instantiation_error`
`Forecaster` is not a valid intermittent-demand forecaster:
`domain_error(forecaster,Forecaster)`
`Observation` is neither a variable nor a number:
`type_error(number,Observation)`
`Observation` is negative:
`domain_error(non_negative_number,Observation)`
Update, error-total, or diagnostic validation arithmetic raises an evaluation error:
`evaluation_error(Error)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

replay_series/5

Replays an already validated series using a concrete method, validated parameters, and valid starting state. Emits pre-observation predictions for numeric targets and fresh placeholders for missing targets, preserving order and all input variables.

Compilation flags:
`static`

Template:
`replay_series(Series,Method,Parameters,State,Values)`
Mode and number of proofs:
`replay_series(@list,+atom,+compound,+compound,-list)` - `one_or_error`

Exceptions:
Prediction or transition arithmetic raises an evaluation error:
`evaluation_error(Error)`

prepare_learning/4

Collects and validates once, fits all requested configurations using exact RMSE priority, and optionally returns aligned predictions without retaining history in the model.

Compilation flags:

`static`

Template:

`prepare_learning(Dataset,Forecaster,Options,Output)`

Mode and number of proofs:

`prepare_learning(+object_identifier,-compound,+list(compound),+compound)` - `one_or_error`

Exceptions:

An observation index or the declared series length is a variable:

`instantiation_error`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The declared series length is not an integer:

`type_error(integer,DeclaredLength)`

The declared series length is not positive:

`domain_error(positive_integer,DeclaredLength)`

The dataset contains no observations:

`domain_error(non_empty_series,Dataset)`

The indices are not a gap-free 1-based sequence:

`domain_error(series_index_sequence,Dataset)`

The declared and observed lengths differ:

`consistency_error(series_length,DeclaredLength,ObservedLength)`

An observation is neither a variable nor a number:

`type_error(number,Observation)`

An observation is negative:

`domain_error(non_negative_number,Observation)`

All observations are missing:

`domain_error(insufficient_observations,Dataset)`

Learning or error-total arithmetic raises an evaluation error:

`evaluation_error(Error)`

normalize_coefficient_grid/2

Normalizes a validated nonempty coefficient grid to ascending distinct floats without modifying the supplied list. Integer 1 and float 1.0 become one candidate. No comparison tolerance is used.

Compilation flags:

```
static
```

Template:

```
normalize_coefficient_grid(Grid,NormalizedGrid)
```

Mode and number of proofs:

```
normalize_coefficient_grid(+list(number),-list(float)) - one_or_error
```

Exceptions:

Coefficient conversion raises an evaluation error:

```
evaluation_error(Error)
```

---

select_methods/7

Evaluates every method while retaining only the best concrete fit.

Compilation flags:

```
static
```

Template:

```
select_methods(Methods,Alphas,Betas,Series,Policy,PreviousBest,Best)
```

Mode and number of proofs:

```
select_methods(+list,+list,+list,@list,+atom,+term,-compound) - one_or_error
```

Exceptions:

Candidate arithmetic raises an evaluation error:

```
evaluation_error(Error)
```

---

select_alphas/7

Evaluates each size coefficient for a concrete method.

Compilation flags:

```
static
```

Template:

```
select_alphas(Alphas,Method,Betas,Series,Policy,PreviousBest,Best)
```

Mode and number of proofs:

```
select_alphas(+list,+atom,+list,@list,+atom,+term,-compound) - one_or_error
```

Exceptions:
Candidate arithmetic raises an evaluation error:
`evaluation_error(Error)`

`select_betas/7`

Fits every occurrence coefficient without suppressing losing-candidate errors.

Compilation flags:
`static`

Template:
`select_betas(Betas,Method,Alpha,Series,Policy,PreviousBest,Best)`
Mode and number of proofs:
`select_betas(+list,+atom,+number,@list,+atom,+term,-compound)` - `one_or_error`

Exceptions:
Candidate arithmetic raises an evaluation error:
`evaluation_error(Error)`

`fit_paired_series/4`

Fits SBA and Croston with shared transitions and independent error totals and metrics.

Compilation flags:
`static`

Template:
`fit_paired_series(Series,Parameters,SBA,Croston)`
Mode and number of proofs:
`fit_paired_series(@list,+compound,-compound,-compound)` - `one_or_error`

Exceptions:
Transition or scoring arithmetic raises an evaluation error:
`evaluation_error(Error)`

---

learn_pair/4

Advances one shared Croston state and two causal error streams.

Compilation flags:
`static`

Template:
`learn_pair(Series,Parameters,Accumulator,Result)`
Mode and number of proofs:
`learn_pair(@list,+compound,+compound,-compound)` - `one_or_error`

Exceptions:
Transition or scoring arithmetic raises an evaluation error:
`evaluation_error(Error)`

---

paired_predictions/4

Computes both method predictions from one rate, retaining integer cold-start zeros.

Compilation flags:
`static`

Template:
`paired_predictions(State,Parameters,SBA,Croston)`
Mode and number of proofs:
`paired_predictions(+compound,+compound,-number,-number)` - `one_or_error`

Exceptions:
Prediction arithmetic raises an evaluation error:
`evaluation_error(Error)`

---

better_candidate/3

Chooses the lower rounded RMSE, breaking exact ties by method, alpha, and beta priority.

Compilation flags:
`static`

Template:
`better_candidate(Candidate,PreviousBest,Best)`
Mode and number of proofs:
`better_candidate(+compound,+term,-compound)` - `one`

---

candidate_key/2

Returns the exact RMSE and deterministic method/coefficient priority key.

Compilation flags:
```
static
```

Template:
```
candidate_key(Candidate,Key)
```
Mode and number of proofs:
```
candidate_key(+compound,-compound) - one
```

---

method_priority/2

Returns the priority of a validated concrete method.

Compilation flags:
```
static
```

Template:
```
method_priority(Method,Priority)
```
Mode and number of proofs:
```
method_priority(+atom,-integer) - one
```

---

fit_series/4

Fits a validated series, retaining terminal state and error metrics without constructing public diagnostics.

Compilation flags:
```
static
```

Template:
```
fit_series(Series,Method,Parameters,Fit)
```
Mode and number of proofs:
```
fit_series(@list,+atom,+compound,-compound) - one_or_error
```

Exceptions:
Fitting or metric arithmetic raises an evaluation error:
```
evaluation_error(Error)
```

---

fit_series_output/5

Fits and scores a validated series, optionally emitting pre-observation predictions in the same traversal.

Compilation flags:
`static`

Template:
`fit_series_output(Series,Method,Parameters,Output,Fit)`
Mode and number of proofs:
`fit_series_output(@list,+atom,+compound,+term,-compound)` - `one_or_error`

Exceptions:
Fitting or metric arithmetic raises an evaluation error:
`evaluation_error(Error)`

---

fitted_model/3

Builds canonical public diagnostics once from the winning fit and validated observation summary.

Compilation flags:
`static`

Template:
`fitted_model(Summary,Fit,Forecaster)`
Mode and number of proofs:
`fitted_model(+compound,+compound,-compound)` - `one_or_error`

Exceptions:
Counter arithmetic raises an evaluation error:
`evaluation_error(Error)`

---

learn_state/6

Advances causal state and error totals with optional aligned prediction output.

Compilation flags:
`static`

Template:
`learn_state(Series,Method,Parameters,Accumulator,Result,Output)`
Mode and number of proofs:
`learn_state(@list,+atom,+compound,+compound,-compound,+term)` - `one_or_error`

Exceptions:

Prediction, transition, or scoring arithmetic raises an evaluation error:

```
evaluation_error(Error)
```

---

score_observation/7

Scores a validated numeric target or preserves totals and a fresh prediction placeholder for a missing target.

Compilation flags:

```
static
```

Template:

```
score_observation(Method,State,Parameters,Observation,PreviousTotals,Totals,
Prediction)
```

Mode and number of proofs:

```
score_observation(+atom,+compound,+compound,@term,+compound,-compound,-term) -
one_or_error
```

Exceptions:

Prediction or scoring arithmetic raises an evaluation error:

```
evaluation_error(Error)
```

---

demand_summary/3

Validates demands and summarizes elapsed, observed, and positive counts without binding missing values.

Compilation flags:

```
static
```

Template:

```
demand_summary(Series,Accumulator,Summary)
```

Mode and number of proofs:

```
demand_summary(@list,+compound,-compound) - one_or_error
```

Exceptions:

An observation is not a number or variable:

```
type_error(number,Observation)
```

An observation is negative:

```
domain_error(non_negative_number,Observation)
```

Counter arithmetic raises an evaluation error:

```
evaluation_error(Error)
```

---

prepare_parameters/4

Prepares validated coefficients, their complements, and the SBA correction for a fitting pass or public operation.

Compilation flags:
`static`

Template:
`prepare_parameters(Alpha,Beta,Policy,Parameters)`
Mode and number of proofs:
`prepare_parameters(+number,+number,+atom,-compound)` - `one_or_error`

Exceptions:
Coefficient arithmetic raises an evaluation error:
`evaluation_error(Error)`

---

positive_smoothing/4

Smooths positive endpoints using their minimum plus a weighted difference, avoiding independent product underflow.

Compilation flags:
`static`

Template:
`positive_smoothing(Weight,Observation,Previous,Smoothed)`
Mode and number of proofs:
`positive_smoothing(+compound,+number,+number,-number)` - `one_or_error`

Exceptions:
Smoothing arithmetic raises an evaluation error:
`evaluation_error(Error)`

---

checked_forecaster/2

Validates a model and returns its extracted fields for reuse by public operations.

Compilation flags:
`static`

Template:
`checked_forecaster(Forecaster,Data)`
Mode and number of proofs:
`checked_forecaster(@term,-compound)` - `one_or_error`

Exceptions:

`Forecaster` is a variable:
`instantiation_error`
`Forecaster` is invalid:
`domain_error(forecaster,Forecaster)`
Validation arithmetic raises an evaluation error:
`evaluation_error(Error)`

`valid_diagnostics/4`

Extracts required fields once and validates options, counters, metrics, and state.

Compilation flags:
`static`

Template:
`valid_diagnostics(Method,State,Diagnostics,Data)`
Mode and number of proofs:
`valid_diagnostics(+atom,+compound,+list,-compound)` - `zero_or_one_or_error`

Exceptions:
Validation arithmetic raises an evaluation error:
`evaluation_error(Error)`

`extract_diagnostics/2`

Extracts ground diagnostic fields, rejecting malformed arities and duplicate error metrics.

Compilation flags:
`static`

Template:
`extract_diagnostics(Diagnostics,Data)`
Mode and number of proofs:
`extract_diagnostics(+list,+compound)` - `zero_or_one`

---

rewrite_diagnostics/4

Rebuilds diagnostic values in one traversal, preserving order, extra metadata, and later common-field occurrences.

Compilation flags:
`static`

Template:
`rewrite_diagnostics(Diagnostics,Data,Seen,UpdatedDiagnostics)`
Mode and number of proofs:
`rewrite_diagnostics(+list,+compound,+non_negative_integer,-list) - one`

---

diagnostic_position/2

Maps required diagnostic names to their compact field positions.

Compilation flags:
`static`

Template:
`diagnostic_position(Name,Position)`
Mode and number of proofs:
`diagnostic_position(+atom,-positive_integer) - zero_or_one`

---

valid_error_metrics/5

Checks numeric error totals and exact consistency of their derived metrics.

Compilation flags:
`static`

Template:
`valid_error_metrics(Count,Absolute,Squared,MAE,RMSE)`
Mode and number of proofs:
`valid_error_metrics(+positive_integer,@term,@term,@term,@term) - zero_or_one_or_error`

Exceptions:
Metric arithmetic raises an evaluation error:
`evaluation_error(Error)`

Operators

(none)

1.153 intervals

object

1.153.1 `interval`

Basic temporal interval relations. An interval is represented by a compound term, `i/2`, with two ground arguments, the start and end points.

Availability:
`logtalk_load(intervals(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2026-05-13

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *interval_protocol*
Aliases:
`interval_protocol` `before/2` as `b/2`
`interval_protocol` `after/2` as `bi/2`
`interval_protocol` `meets/2` as `m/2`
`interval_protocol` `met_by/2` as `mi/2`
`interval_protocol` `overlaps/2` as `o/2`
`interval_protocol` `overlapped_by/2` as `oi/2`
`interval_protocol` `starts/2` as `s/2`
`interval_protocol` `started_by/2` as `si/2`
`interval_protocol` `during/2` as `d/2`
`interval_protocol` `contains/2` as `di/2`
`interval_protocol` `finishes/2` as `f/2`
`interval_protocol` `finished_by/2` as `fi/2`
`interval_protocol` `equal/2` as `eq/2`

Remarks:
(none)

Inherited public predicates:
after/2 *before/2* *contains/2* *during/2* *equal/2* *finished_by/2* *finishes/2* *meets/2* *met_by/2* *new/3* *overlapped_by/2* *overlaps/2* *relation/3* *started_by/2* *starts/2* *valid/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.153.2 `interval_algebra`

Allen interval algebra over the 13 base relation atoms. Composition results are returned as canonical ordered duplicate-free lists.

Availability:
`logtalk_load(intervals(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-13

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *interval_algebra_protocol*

Remarks:
(none)

Inherited public predicates:
compose/3 *converse/2* *relation/1*

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.153.3 `interval_algebra_protocol`

Allen interval algebra protocol over the 13 base relation atoms.

Availability:
`logtalk_load(intervals(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *relation/1*
  - *converse/2*
  - *compose/3*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

relation/1

Enumerates valid Allen base relation atoms in canonical order.

Compilation flags:
`static`

Template:
`relation(Relation)`
Mode and number of proofs:
`relation(?atom)` - `zero_or_more`

---

converse/2

Relates an Allen base relation atom with its converse relation atom.

Compilation flags:
`static`

Template:
`converse(Relation,Converse)`
Mode and number of proofs:
`converse(?atom,?atom)` - `zero_or_more`

---

compose/3

Returns the canonical ordered list of Allen base relation atoms resulting from composing two Allen base relation atoms.

Compilation flags:
`static`

Template:
`compose(Relation1,Relation2,Relations)`
Mode and number of proofs:
`compose(+atom,+atom,-list(atom))` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

interval_protocol, *interval*, *interval_algebra*, *interval_relation_set_protocol*, *interval_relation_set*

object

1.153.4 `interval_constraint_network`

Allen interval constraint-network operations built on canonical relation sets, intended for small-to-medium symbolic networks.

Availability:
`logtalk_load(intervals(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *interval_constraint_network_protocol*
Uses:
interval_relation_set
list

Remarks:
(none)

Inherited public predicates:
consistent/1 *constraints/2* *contradiction/2* *contradiction_explanations/2* *entailment_explanations/5* *entails/4* *entails/5* *equivalent/2* *excluded/4* *network/1* *new/2* *nodes/2* *path_consistency/2* *possible/4* *propagate/2* *propagate/3* *propagation_triple/2* *propagation_triples/2* *refine/5* *refine_propagate/3* *refine_propagate/4* *refine_propagate/5* *refine_propagate/6* *relation/4* *subsumes/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.153.5 `interval_constraint_network_protocol`

Allen interval constraint-network protocol built on canonical relation sets, intended for small-to-medium symbolic networks.

Availability:
`logtalk_load(intervals(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *network/1*
  - *new/2*
  - *nodes/2*
  - *relation/4*
  - *entails/4*
  - *entails/5*
  - *possible/4*
  - *excluded/4*

- *contradiction/2*
- *entailment_explanations/5*
- *contradiction_explanations/2*
- *refine/5*
- *consistent/1*
- *propagate/2*
- *propagate/3*
- *refine_propagate/5*
- *refine_propagate/6*
- *refine_propagate/3*
- *refine_propagate/4*
- *propagation_triple/2*
- *propagation_triples/2*
- *path_consistency/2*
- *constraints/2*
- *subsumes/2*
- *equivalent/2*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

network/1

True if Network is a valid interval constraint network represented as a network(Nodes, Constraints) term.

Compilation flags:

```
static
```

Template:

```
network(Network)
```

Mode and number of proofs:

```
network(?compound) - zero_or_more
```

---

new/2

Constructs a new network for a list of distinct nodes. Distinct node pairs are initialized with the universal relation set.

Compilation flags:

```
static
```

Template:

```
new(Nodes,Network)
```

Mode and number of proofs:

```
new(+list,-compound) - zero_or_one
```

nodes/2

Returns the list of nodes in a network.

Compilation flags:
`static`

Template:
`nodes(Network,Nodes)`
Mode and number of proofs:
`nodes(+compound,-list)` - `zero_or_one`

---

relation/4

Returns the current relation set between two nodes in a network. A node is always equal to itself.

Compilation flags:
`static`

Template:
`relation(Network,Node1,Node2,RelationSet)`
Mode and number of proofs:
`relation(+compound,@term,@term,-list(atom))` - `zero_or_one`

---

entails/4

True if the current relation set between two nodes is a subset of a queried relation set.

Compilation flags:
`static`

Template:
`entails(Network,Node1,Node2,RelationSet)`
Mode and number of proofs:
`entails(+compound,@term,@term,+list(atom))` - `zero_or_one`

---

entails/5

True if the current relation set between two nodes is a subset of a queried relation set and returns an explanation term describing either a direct constraint or a supporting propagation path.

Compilation flags:

```
static
```

Template:

```
entails(Network,Node1,Node2,RelationSet,Explanation)
```

Mode and number of proofs:

```
entails(+compound,@term,@term,+list(atom),-compound) - zero_or_one
```

---

possible/4

True if the current relation set between two nodes intersects a queried relation set.

Compilation flags:

```
static
```

Template:

```
possible(Network,Node1,Node2,RelationSet)
```

Mode and number of proofs:

```
possible(+compound,@term,@term,+list(atom)) - zero_or_one
```

---

excluded/4

True if the current relation set between two nodes is disjoint from a queried relation set.

Compilation flags:

```
static
```

Template:

```
excluded(Network,Node1,Node2,RelationSet)
```

Mode and number of proofs:

```
excluded(+compound,@term,@term,+list(atom)) - zero_or_one
```

---

contradiction/2

Returns a simple contradiction explanation for an inconsistent network as contradiction(Node1, Node2, Cause).

Compilation flags:

```
static
```

Template:

```
contradiction(Network,Explanation)
```

Mode and number of proofs:

```
contradiction(+compound,-compound) - zero_or_more
```

---

entailment_explanations/5

Returns the list of immediate explanations supporting an entailed relation query, using identity, direct, and propagated explanation terms.

Compilation flags:

```
static
```

Template:

```
entailment_explanations(Network,Node1,Node2,RelationSet,Explanations)
```

Mode and number of proofs:

```
entailment_explanations(+compound,@term,@term,+list(atom),-list(compound)) -
zero_or_one
```

---

contradiction_explanations/2

Returns the list of contradiction explanations for all empty explicit pair constraints in a network.

Compilation flags:

```
static
```

Template:

```
contradiction_explanations(Network,Explanations)
```

Mode and number of proofs:

```
contradiction_explanations(+compound,-list(compound)) - zero_or_one
```

---

refine/5

Refines the relation set between two distinct nodes by intersecting it with a new relation set.

Compilation flags:
```
static
```

Template:
```
refine(Network,Node1,Node2,RelationSet,RefinedNetwork)
```
Mode and number of proofs:
```
refine(+compound,@term,@term,+list(atom),-compound) - zero_or_one
```

---

consistent/1

True if all explicit pair constraints in a network are non-empty.

Compilation flags:
```
static
```

Template:
```
consistent(Network)
```
Mode and number of proofs:
```
consistent(+compound) - zero_or_one
```

---

propagate/2

Computes a path-consistency closure and succeeds only if the resulting network is consistent. Intended for small-to-medium symbolic networks; not a large-scale temporal CSP solver.

Compilation flags:
```
static
```

Template:
```
propagate(Network,Closure)
```
Mode and number of proofs:
```
propagate(+compound,-compound) - zero_or_one
```

---

propagate/3

Computes a path-consistency closure, returns the list of propagated changes, and succeeds only if the resulting network is consistent. Intended for small-to-medium symbolic networks; not a large-scale temporal CSP solver.

Compilation flags:

```
static
```

Template:

```
propagate(Network,Closure,Changes)
```

Mode and number of proofs:

```
propagate(+compound,-compound,-list(compound)) - zero_or_one
```

---

refine_propagate/5

Refines the relation set between two distinct nodes and then propagates the consequences, succeeding only if the resulting network is consistent. Intended for small-to-medium symbolic networks; not a large-scale temporal CSP solver.

Compilation flags:

```
static
```

Template:

```
refine_propagate(Network,Node1,Node2,RelationSet,Closure)
```

Mode and number of proofs:

```
refine_propagate(+compound,@term,@term,+list(atom),-compound) - zero_or_one
```

---

refine_propagate/6

Refines the relation set between two distinct nodes, propagates the consequences incrementally, and returns the list of direct and propagated changes, succeeding only if the resulting network is consistent. Intended for small-to-medium symbolic networks; not a large-scale temporal CSP solver.

Compilation flags:

```
static
```

Template:

```
refine_propagate(Network,Node1,Node2,RelationSet,Closure,Changes)
```

Mode and number of proofs:

```
refine_propagate(+compound,@term,@term,+list(atom),-compound,-list(compound)) -
zero_or_one
```

---

refine_propagate/3

Applies a batch of relation-set refinements represented as constraint(Node1, Node2, RelationSet) terms and then propagates the combined consequences, succeeding only if the resulting network is consistent. Intended for small-to-medium symbolic networks; not a large-scale temporal CSP solver.

Compilation flags:

```
static
```

Template:

```
refine_propagate(Network,Constraints,Closure)
```

Mode and number of proofs:

```
refine_propagate(+compound,+list(compound),-compound) - zero_or_one
```

---

refine_propagate/4

Applies a batch of relation-set refinements, propagates the combined consequences incrementally, and returns the list of direct and propagated changes, succeeding only if the resulting network is consistent. Intended for small-to-medium symbolic networks; not a large-scale temporal CSP solver.

Compilation flags:

```
static
```

Template:

```
refine_propagate(Network,Constraints,Closure,Changes)
```

Mode and number of proofs:

```
refine_propagate(+compound,+list(compound),-compound,-list(compound)) -
zero_or_one
```

---

propagation_triple/2

Extracts the causing node triple triple(Node1, Node3, Node2) from a propagated change term.

Compilation flags:

```
static
```

Template:

```
propagation_triple(Change,Triple)
```

Mode and number of proofs:

```
propagation_triple(+compound,-compound) - zero_or_one
```

---

propagation_triples/2

Collects the unique node triples that caused propagated refinements in a list of change terms.

Compilation flags:
`static`

Template:
`propagation_triples(Changes,Triples)`
Mode and number of proofs:
`propagation_triples(+list(compound),-list(compound))` - `zero_or_one`

---

path_consistency/2

Computes the path-consistency closure of a network by iterative refinement using relation-set composition and intersection. Intended for small-to-medium symbolic networks; not a large-scale temporal CSP solver.

Compilation flags:
`static`

Template:
`path_consistency(Network,Closure)`
Mode and number of proofs:
`path_consistency(+compound,-compound)` - `zero_or_one`

---

constraints/2

Returns the explicit pair constraints in canonical storage order using constraint(Node1, Node2, RelationSet) terms.

Compilation flags:
`static`

Template:
`constraints(Network,Constraints)`
Mode and number of proofs:
`constraints(+compound,-list(compound))` - `zero_or_one`

---

subsumes/2

True if the first network is at least as general as the second network, i.e. every pair relation set in the second network is a subset of the corresponding relation set in the first network.

Compilation flags:

```
static
```

Template:

```
subsumes(Network1,Network2)
```

Mode and number of proofs:

```
subsumes(+compound,+compound) - zero_or_one
```

---

equivalent/2

True if two networks contain the same nodes and mutually subsume each other.

Compilation flags:

```
static
```

Template:

```
equivalent(Network1,Network2)
```

Mode and number of proofs:

```
equivalent(+compound,+compound) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

interval_relation_set_protocol, *interval_relation_set*, *interval_algebra_protocol*, *interval_algebra*

protocol

1.153.6 interval_protocol

Basic temporal interval relations protocol (based on James F. Allen Interval Algebra work).

Availability:
`logtalk_load(intervals(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-05-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *new/3*
  - *valid/1*
  - *relation/3*
  - *before/2*
  - *after/2*
  - *meets/2*
  - *met_by/2*
  - *overlaps/2*
  - *overlapped_by/2*
  - *starts/2*
  - *started_by/2*
  - *during/2*
  - *contains/2*
  - *finishes/2*
  - *finished_by/2*
  - *equal/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new/3

Constructs a new interval given start and end points. The start point must strictly precede the end point.

Compilation flags:
`static`

Template:
`new(Start,End,Interval)`
Mode and number of proofs:
`new(@ground,@ground,-interval)` - `zero_or_one`

---

valid/1

True if Interval is a valid interval.

Compilation flags:
`static`

Template:
`valid(Interval)`
Mode and number of proofs:
`valid(@interval)` - `zero_or_one`

---

relation/3

Classifies two valid intervals according to their unique Allen base relation.

Compilation flags:
`static`

Template:
`relation(Interval1,Interval2,Relation)`
Mode and number of proofs:
`relation(@interval,@interval,?atom)` - `zero_or_one`

---

before/2

True if Interval1 takes place before Interval2.

Compilation flags:
static

Template:
before(Interval1,Interval2)
Mode and number of proofs:
before(@interval,@interval) - zero_or_one

---

after/2

True if Interval1 takes place after Interval2.

Compilation flags:
static

Template:
after(Interval1,Interval2)
Mode and number of proofs:
after(@interval,@interval) - zero_or_one

---

meets/2

True if Interval1 meets Interval2.

Compilation flags:
static

Template:
meets(Interval1,Interval2)
Mode and number of proofs:
meets(@interval,@interval) - zero_or_one

---

met_by/2

True if Interval1 is met by Interval2.

Compilation flags:
static

Template:
met_by(Interval1,Interval2)
Mode and number of proofs:
met_by(@interval,@interval) - zero_or_one

---

overlaps/2

True if Interval1 overlaps with Interval2.

Compilation flags:
static

Template:
overlaps(Interval1,Interval2)
Mode and number of proofs:
overlaps(@interval,@interval) - zero_or_one

---

overlapped_by/2

True if Interval1 is overlapped by Interval2.

Compilation flags:
static

Template:
overlapped_by(Interval1,Interval2)
Mode and number of proofs:
overlapped_by(@interval,@interval) - zero_or_one

---

starts/2

True if `Interval1` starts `Interval2`.

Compilation flags:
`static`

Template:
`starts(Interval1,Interval2)`
Mode and number of proofs:
`starts(@interval,@interval)` - `zero_or_one`

---

started_by/2

True if `Interval1` is started by `Interval2`.

Compilation flags:
`static`

Template:
`started_by(Interval1,Interval2)`
Mode and number of proofs:
`started_by(@interval,@interval)` - `zero_or_one`

---

during/2

True if `Interval1` occurs during `Interval2`.

Compilation flags:
`static`

Template:
`during(Interval1,Interval2)`
Mode and number of proofs:
`during(@interval,@interval)` - `zero_or_one`

---

contains/2

True if `Interval1` contains `Interval2`.

Compilation flags:
`static`

Template:
`contains(Interval1,Interval2)`
Mode and number of proofs:
`contains(@interval,@interval)` - `zero_or_one`

---

finishes/2

True if `Interval1` finishes `Interval2`.

Compilation flags:
`static`

Template:
`finishes(Interval1,Interval2)`
Mode and number of proofs:
`finishes(@interval,@interval)` - `zero_or_one`

---

finished_by/2

True if `Interval1` is finished by `Interval2`.

Compilation flags:
`static`

Template:
`finished_by(Interval1,Interval2)`
Mode and number of proofs:
`finished_by(@interval,@interval)` - `zero_or_one`

`equal/2`

True if `Interval1` is equal to `Interval2`.

Compilation flags:
`static`

Template:
`equal(Interval1,Interval2)`
Mode and number of proofs:
`equal(@interval,@interval)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

interval, *interval_algebra_protocol*, *interval_algebra*, *interval_relation_set_protocol*, *interval_relation_set*

object

1.153.7 `interval_relation_set`

Allen interval relation-set operations using canonical ordered duplicate-free lists of base relation atoms.

Availability:
`logtalk_load(intervals(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *interval_relation_set_protocol*
Uses:
interval_algebra
list

Remarks:
(none)

Inherited public predicates:
compose/3 *converse/2* *empty/1* *intersection/3* *member/2* *normalize/2* *relation_set/1* *singleton/2* *subset/2* *union/3* *universal/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.153.8 `interval_relation_set_protocol`

Allen interval relation-set protocol using canonical ordered duplicate-free lists of base relation atoms.

Availability:
`logtalk_load(intervals(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *relation_set/1*
  - *empty/1*
  - *universal/1*
  - *singleton/2*
  - *normalize/2*
  - *member/2*
  - *subset/2*
  - *intersection/3*
  - *union/3*
  - *converse/2*
  - *compose/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

relation_set/1

True if RelationSet is a canonical ordered duplicate-free list of Allen base relation atoms.

Compilation flags:
`static`

Template:
`relation_set(RelationSet)`
Mode and number of proofs:
`relation_set(?list(atom))` - `zero_or_more`

---

empty/1

Returns the empty Allen relation set.

Compilation flags:
`static`

Template:
`empty(RelationSet)`
Mode and number of proofs:
`empty(?list(atom))` - `one`

---

universal/1

Returns the universal Allen relation set containing all 13 base relation atoms.

Compilation flags:
`static`

Template:
`universal(RelationSet)`
Mode and number of proofs:
`universal(?list(atom))` - `one`

---

singleton/2

Relates an Allen base relation atom with its singleton relation set.

Compilation flags:
`static`

Template:
`singleton(Relation,RelationSet)`
Mode and number of proofs:
`singleton(?atom,?list(atom))` - `zero_or_more`

---

normalize/2

Normalizes a list of Allen base relation atoms into canonical ordered duplicate-free form.

Compilation flags:
`static`

Template:
`normalize(Relations,RelationSet)`
Mode and number of proofs:
`normalize(+list(atom),-list(atom))` - `zero_or_one`

---

member/2

True if Relation is a member of RelationSet.

Compilation flags:
`static`

Template:
`member(Relation,RelationSet)`
Mode and number of proofs:
`member(?atom,+list(atom))` - `zero_or_more`

---

subset/2

True if RelationSet1 is a subset of RelationSet2.

Compilation flags:
`static`

Template:
`subset(RelationSet1,RelationSet2)`
Mode and number of proofs:
`subset(+list(atom),+list(atom))` - `zero_or_one`

---

intersection/3

Computes the intersection of two relation sets.

Compilation flags:
`static`

Template:
`intersection(RelationSet1,RelationSet2,Intersection)`
Mode and number of proofs:
`intersection(+list(atom),+list(atom),-list(atom))` - `zero_or_one`

---

union/3

Computes the union of two relation sets.

Compilation flags:
```
static
```

Template:
```
union(RelationSet1,RelationSet2,Union)
```
Mode and number of proofs:
```
union(+list(atom),+list(atom),-list(atom)) - zero_or_one
```

---

converse/2

Maps a relation set to the canonical ordered set of converse relations.

Compilation flags:
```
static
```

Template:
```
converse(RelationSet,ConverseSet)
```
Mode and number of proofs:
```
converse(+list(atom),-list(atom)) - zero_or_one
```

---

compose/3

Computes the canonical ordered relation set resulting from composing two relation sets.

Compilation flags:
```
static
```

Template:
```
compose(RelationSet1,RelationSet2,Composition)
```
Mode and number of proofs:
```
compose(+list(atom),+list(atom),-list(atom)) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

interval_algebra_protocol, *interval_algebra*, *interval_relation_set*

1.154 iqr_anomaly_detector

object

1.154.1 `iqr_anomaly_detector`

Statistical interquartile-range anomaly detector for continuous datasets. Learns per-attribute quartiles and interquartile ranges from baseline training examples selected from a dataset object implementing the `anomaly_dataset_protocol` protocol and returns a detector term that can be used for scoring, prediction, and export.

Availability:

```
logtalk_load(iqr_anomaly_detector(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public anomaly_detector_common
```

Uses:

format
list
numberlist
pairs
sample
type

Remarks:

(none)

Inherited public predicates:

anomaly_detector_options/2 *check_anomaly_detector/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict/4* *print_anomaly_detector/1* *score/3* *score_all/3* *valid_anomaly_detector/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

anomaly_dataset_protocol, *anomaly_detector_protocol*, *modified_z_score_anomaly_detector*, *z_score_anomaly_detector*, *isolation_forest_anomaly_detector*, *knn_distance_anomaly_detector*, *lof_anomaly_detector*

1.155 iso8601

object

1.155.1 `iso8601`

ISO 8601 (and European civil calendar) compliant library of date and time predicates.

Availability:

```
logtalk_load(iso8601(loader))
```

Author: Daniel L. Dudley and Paulo Moura
Version: 1:4:0
Date: 2026-08-12

Compilation flags:

```
static, context_switching_calls
```

Uses:

list
os

Remarks:

- Scope: This object currently provides a powerful, versatile and efficient set of date-handling predicates, which–thanks to Logtalk–may be used as is on a wide range of Prolog compilers. Besides taking time to familiarize oneself with each predicate, the user should take note of the following information.
- Validation of dates: Date parts are not validated–that is the caller's responsibility! However, not being quite heartless yet, we do provide a predicate for this purpose.
- Date arithmetic: Many of the examples illustrate a simplified method of doing date arithmetic. Note, however, that we do not generally recommend this practice–it is all too easy to make mistakes. The safest way of finding the day difference between two dates is to first convert the dates to their Julian day numbers and then subtract one from the other. Similarly, the safe way to add or subtract a day offset to a particular date is to first convert the date to its Julian day number, add or subtract the day offset, and then convert the result to its corresponding date.
- BC years: ISO 8601 specifies that the Gregorian calendar be used, yet requires that years prior to 1 AD be handled arithmetically, i.e., the year we know as 1 BC is year 0, 2 BC is year -1, 3 BC is year -2 and so on. We do not follow ISO 8601 with regard to the handling of BC years. Our date predicates will accept and interpret an input year 0 as 1 BC; however, a negative year, Year, should always be interpreted as `abs(Year) =:= Year BC`. We believe that the average person will find our handling of BC years more user-friendly than the ISO 8601 one, but we encourage feedback from users with a view to a possible change in future versions.
- Week numbers: It is possible for a day (date) to have a week number that belongs to another year. Up to three of the first days of a calendar year may belong to the last week (number) of the prior calendar year, and up to three days of the last days of a calendar year may belong to the first week (number) of the next calendar year. It for this reason that the Week parameter in `date/6-7` is a compound term, namely `week(WeekNo,ActualYear)`.
- Computation of Gregorian Easter Sunday: The algorithm is based upon the "Gaussian rule". Proleptic use is limited to years > 1582 AD, that is, after the introduction of the Gregorian calendar.
- Some Christian feast day offsets from Easter Sunday: Carnival Monday: -48 days, Mardi Gras (Shrove Tuesday): -47 days, Ash Wednesday: -46 days, Palm Sunday: -7 days, Easter Friday: -2 days, Easter Saturday: -1 day, Easter Monday: +1 day, Ascension of Christ: +39 days, Whitsunday: +49 days, Whitmonday: +50 days, Feast of Corpus Christi: +60 days.

Inherited public predicates:

(none)

- *Public predicates*
  - *date/4*
  - *date/5*
  - *date/6*
  - *date/7*
  - *date_string/3*
  - *time_string/3*
  - *date_time_string/3*
  - *duration_string/2*
  - *interval_string/2*
  - *valid_date/3*
  - *leap_year/1*
  - *calendar_month/3*
  - *easter_day/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

date/4

Get the system date and/or its Julian Day # or convert a Julian Day # to/from given date parts.

Compilation flags:

`static`

Template:

`date(JD,Year,Month,Day)`

`JD` - Julian day serial number.

`Year` - 0 or negative if converted BC year, positive otherwise.

`Month` - Normally an integer between 1 and 12 inclusive.

`Day` - Normally an integer between 1 and 31 inclusive depending upon month.

Mode and number of proofs:

`date(?integer,?integer,?integer,?integer)` - `zero_or_one`

Examples:

Current date (i.e., today)

```
date(JD,Year,Month,Day)
JD=2453471,Year=2005,Month=4,Day=10
```

Convert a date to its Julian day number

```
date(JD,2000,2,29)
JD=2451604
```

Convert a Julian day number to its date

```
date(2451604,Year,Month,Day)
Year=2000,Month=2,Day=29
```

What is the date of day # 60 in year 2000?

```
date(JD,2000,1,60)
JD=2451604
```

What is the Julian of the 1st day prior to 2000-1-1?

```
date(JD,2000,1,0)
JD=2451544
```

What is the Julian of the 60th day prior to 2000-1-1?

```
date(JD,2000,1,-59)
JD=2451485
```

Illegal date is auto-adjusted (see also next query)

```
date(JD,1900,2,29)
JD=2415080
```

This is the correct date!

```
date(2415080,Year,Month,Day)
Year=1900,Month=3,Day=1
```

date/5

Ditto `date/4` + get/check its day-of-week #.

Compilation flags:

`static`

Template:

`date(JD,Year,Month,Day,DoW)`

`JD` - Julian day serial number.

`Year` - 0 or negative if converted BC year, positive otherwise.

`Month` - Normally an integer between 1 and 12 inclusive.

`Day` - Normally an integer between 1 and 31 inclusive depending upon month.

`DoW` - Day of week, where Monday=1, Tuesday=2, …, Sunday=7.

Mode and number of proofs:

`date(?integer,?integer,?integer,?integer,?integer)` - `zero_or_one`

Examples:

Get the Julian and the day-of-week # of a date

```
date(JD,2000,2,29,DoW)
JD=2451604,DoW=2
```

Check the validity of a given date (day-of-week is 2, not 4)

```
date(_,2002,3,5,4)
no
```

Get the Julian day of a given date if it is a Sunday

```
date(JD,2004,2,29,7)
JD=2453065
```

Get the date and day-of-week # of a Julian

```
date(2451545,Year,Month,Day,DoW)
Year=2000,Month=1,Day=1,DoW=6
```

---

date/6

Ditto `date/5` + get/check its week #.

Compilation flags:

`static`

Template:

`date(JD,Year,Month,Day,DoW,Week)`

`JD` - Julian day serial number.

`Year` - 0 or negative if converted BC year, positive otherwise.

`Month` - Normally an integer between 1 and 12 inclusive.

`Day` - Normally an integer between 1 and 31 inclusive depending upon month.

`DoW` - Day of week, where Monday=1, Tuesday=2, …, Sunday=7.

`Week` - Compound term, `week(WeekNo,ActualYear)`, of a day.

Mode and number of proofs:

`date(?integer,?integer,?integer,?integer,?integer,?compound)` - `zero_or_one`

Examples:

Get the day-of-week and week number of a date

```
date(_,2000,1,1,DoW,Week)
DoW=6,Week=week(52,1999)
```

Get the week number and year of this week

```
date(_,_,_,_,_,Week)
Week=week(7,2004)
```

Get the Julian number and the week of a date if it is a Sunday

```
date(JD,2004,2,29,7,Week)
JD=2453065,Week=week(9,2004)
```

Get the day-of-week and week of a Julian day number

```
date(2453066,_,_,_,DoW,Week)
DoW=1,Week=week(10,2004)
```

Check that given date data matches

```
date(_,2004,3,1,1,week(10,2004))
yes
```

What is the date of a day of week (default is 1) in given week # and year?

```
date(_,Year,Month,Day,DoW,week(26,2004))
Year=2004,Month=6,Day=21,DoW=1
```

Ditto for Sunday

```
date(_,Year,Month,Day,7,week(1,2005))
Year=2005,Month=1,Day=9
```

Ditto for Tuesday in following week

```
date(_,Year,Month,Day,9,week(1,2005))
Year=2005,Month=1,Day=11
```

Ditto for Thursday in the prior week

```
date(_,Year,Month,Day,4,week(0,2005))
Year=2004,Month=12,Day=30
```

Ditto for Tuesday two weeks prior

```
date(_,Year,Month,Day,2,week(-1,2005))
Year=2004,Month=12,Day=21
```

Ditto for Saturday

```
date(_,Year,Month,Day,6,week(53,2004))
Year=2005,Month=1,Day=1
```

Ditto for Monday (note automatic compensation of nonexistent week number)

```
date(_,Year,Month,Day,1,week(60,2004))
Year=2005,Month=2,Day=14
```

date/7

Ditto `date/6` + get/check its day-of-year #.

Compilation flags:

`static`

Template:

`date(JD,Year,Month,Day,DoW,Week,DoY)`

`JD` - Julian day serial number.

`Year` - 0 or negative if converted BC year, positive otherwise.

`Month` - Normally an integer between 1 and 12 inclusive.

`Day` - Normally an integer between 1 and 31 inclusive depending upon month.

`DoW` - Day of week, where Monday=1, Tuesday=2, …, Sunday=7.

`Week` - Compound term, `week(WeekNo,ActualYear)`, of a day.

`DoY` - Day of year (NB! calendar year, not week # year).

Mode and number of proofs:

`date(?integer,?integer,?integer,?integer,?integer,?compound,?integer)` - `zero_or_one`

Examples:

Get the date and day-of-year of a Julian number

```
date(2451649,Year,Month,Day,_,_,DoY)
Year=2000,Month=4,Day=14,DoY=105
```

Get the Julian number, week number and day-of-year of a date, confirming that it is a Sunday

```
date(JD,2004,2,29,7,Week,DoY)
JD=2453065,Week=week(9,2004),DoY=60
```

Confirm that a date is, in fact, a specific day-of-year

```
date(_,2004,3,1,_,_,61)
yes
```

Get the Julian number, week day and day-of-year of a date

```
date(JD,2004,10,18,DoW,_,DoY)
JD=2453297,DoW=1,DoY=292
```

Get today's day-of-year

```
date(_,_,_,_,_,_,DoY)
DoY=54
```

Get all missing date data (excl. Julian number) for the 60th calendar day of 2004

```
date(_,2004,Month,Day,DoW,Week,60)
Month=2,Day=29,DoW=7,Week=week(9,2004)
```

Match given date data and, if true, return the missing data (excl. Julian number)

```
date(_,2004,3,Day,DoW,Week,61)
Day=1,DoW=1,Week=week(10,2004)
```

Ditto (the 61st day-of-year cannot be both day 1 and 2 of the month)

```
date(_,2004,_,2,_,_,61)
no
```

date_string/3

Conversion between an ISO 8601 compliant date string and its components (truncated and expanded date representations are currently unsupported). Note that date components are not validated; that is the caller's responsibility!

Compilation flags:

`static`

Template:

`date_string(Format,Components,String)`

`Format` - ISO 8601 format.

`Components` - When bound and String is free, either a Julian number or a `[Year,Month,Day]` term; it binds to the system day/date if free When free and String is bound, it binds to an integer list representing the numeric elements of String.

`String` - ISO 8601 formatted string correspondent to Components.

Mode and number of proofs:

`date_string(+atom,+integer,?atom)` - `zero_or_one`

`date_string(+atom,?list,?atom)` - `zero_or_one`

Examples:

Date, complete, basic (section 5.2.1.1)

```
date_string('YYYYMMDD',[2004,2,29],String)
String='20040229'
```

Date, complete, basic (section 5.2.1.1)

```
date_string('YYYYMMDD',Components,'20040229')
Components=[2004,2,29]
```

Date, complete, extended (section 5.2.1.1)

```
date_string('YYYY-MM-DD',[2003,12,16],String)
String='2003-12-16'
```

Date, complete, extended (section 5.2.1.1)

```
date_string('YYYY-MM-DD',Components,'2003-12-16')
Components=[2003,12,16]
```

Date, complete, extended (section 5.2.1.1)

```
date_string('YYYY-MM-DD',_,String)
String='2004-02-17'
```

Date, complete, extended (section 5.2.1.1)

```
date_string('YYYY-MM-DD',Components,'2004-02-17')
Components=[2004,2,17]
```

Date, reduced, month (section 5.2.1.2 a)

```
date_string('YYYY-MM',[2004,9,18],String)
String='2004-09'
```

Date, reduced, month (section 5.2.1.2 a)

```
date_string('YYYY-MM',Components,'2004-09')
Components=[2004,9]
```

Date, reduced, year (section 5.2.1.2 b)

```
date_string('YYYY',[1900,7,24],String)
String='1900'
```

Date, reduced, year (section 5.2.1.2 b)

```
date_string('YYYY',Components,'1900')
Components=[1900]
```

Date, reduced, century (section 5.2.1.2 c)

```
date_string('YY',2456557,String)
String='20'
```

Date, reduced, century (section 5.2.1.2 c)

```
date_string('YY',Components,'20')
Components=[20]
```

Date, ordinal, complete (section 5.2.2.1)

```
date_string('YYYYDDD',[2005,3,25],String)
String='2005084'
```

Date, ordinal, complete (section 5.2.2.1)

```
date_string('YYYYDDD',Components,'2005084')
Components=[2005,84]
```

Date, ordinal, extended (section 5.2.2.1)

```
date_string('YYYY-DDD',[1854,12,4],String)
String='1854-338'
```

Date, ordinal, extended (section 5.2.2.1)

```
date_string('YYYY-DDD',Components,'1854-338')
Components=[1854,338]
```

Week, complete, basic (section 5.2.3.1)

```
date_string('YYYYWwwD',[2000,1,2],String)
String='1999W527'
```

Week, complete, basic (section 5.2.3.1)

```
date_string('YYYYWwwD',Components,'1999W527')
Components=[1999,52,7]
```

Week, complete, extended (section 5.2.3.1)

```
date_string('YYYY-Www-D',[2003,12,29],String)
String='2004-W01-1'
```

Week, complete, extended (section 5.2.3.1)

```
date_string('YYYY-Www-D',Components,'2004-W01-1')
Components=[2004,1,1]
```

Week, complete, extended (section 5.2.3.1)

```
date_string('YYYY-Www-D',2453167,String)
String='2004-W24-4'
```

Week, complete, extended (section 5.2.3.1)

```
date_string('YYYY-Www-D',Components,'2004-W24-4')
Components=[2004,24,4]
```

Week, reduced, basic (section 5.2.3.2)

```
date_string('YYYYWww',[2004,2,29],String)
String='2004W09'
```

Week, reduced, basic (section 5.2.3.2)

```
date_string('YYYYWww',Components,'2004W09')
Components=[2004,9]
```

Week, reduced, extended (section 5.2.3.2)

```
date_string('YYYY-Www',[2004,2,29],String)
String='2004-W09'
```

Week, reduced, extended (section 5.2.3.2)

```
date_string('YYYY-Www',Components,'2004-W09')
Components=[2004,9]
```

time_string/3

Conversion between an ISO 8601 compliant time-of-day string and a `time(Hours,Minutes,Seconds)` term. Supported forms include basic and extended notations, with optional fractional seconds.

Compilation flags:

`static`

Template:

`time_string(Format,Time,String)`

Mode and number of proofs:

`time_string(+atom,?compound,?atom)` - `zero_or_one`

Examples:

Time, complete, extended

```
time_string('Thh:mm:ss',time(14,30,0),String)
String='T14:30:00'
```

Time, complete, basic

```
time_string('Thhmmss',Time,'T143000')
Time=time(14,30,0)
```

Time, complete, extended, fractional seconds

```
time_string('Thh:mm:ss.s',Time,'T14:30:00.125')
Time=time(14,30,0.125)
```

Time, complete, extended, comma fractional seconds

```
time_string('Thh:mm:ss,s',Time,'T14:30:00,125')
Time=time(14,30,0.125)
```

---

date_time_string/3

Conversion between an ISO 8601 compliant combined date-time string and a `date_time/6` or `date_time/7` term. Offset-aware terms use `date_time(Year,Month,Day,Hours,Minutes,Seconds,OffsetSeconds)` where `OffsetSeconds` is the UTC offset in seconds.

Compilation flags:

`static`

Template:

`date_time_string(Format,DateTime,String)`

Mode and number of proofs:

`date_time_string(+atom,?compound,?atom)` - `zero_or_one`

Examples:

Date-time, complete, extended

```
date_time_string('YYYY-MM-DDThh:mm:ss',date_time(2026,4,7,14,30,0),String)
String='2026-04-07T14:30:00'
```

Date-time, ordinal, extended, UTC

```
date_time_string('YYYY-DDDThh:mm:ssZ',DateTime,'2026-097T14:30:00Z')
DateTime=date_time(2026,4,7,14,30,0,0)
```

Date-time, week, complete, extended

```
date_time_string('YYYY-Www-DThh:mm:ss',date_time(2026,4,7,14,30,0),String)
String='2026-W15-2T14:30:00'
```

Date-time, complete, basic, UTC

```
date_time_string('YYYYMMDDThhmmssZ',DateTime,'20260407T143000Z')
DateTime=date_time(2026,4,7,14,30,0,0)
```

Date-time, complete, extended, offset

```
date_time_string('YYYY-MM-DDThh:mm:ss+hh:mm',DateTime,
'2026-04-07T14:30:00+05:45')
DateTime=date_time(2026,4,7,14,30,0,20700)
```

Date-time, complete, extended, comma fractional seconds

```
date_time_string('YYYY-MM-DDThh:mm:ss,sZ',DateTime,'2026-04-07T14:30:00,
125Z')
DateTime=date_time(2026,4,7,14,30,0.125,0)
```

---

duration_string/2

Conversion between an ISO 8601 duration string and a `duration(Years,Months,Days,Hours,Minutes,Seconds)` term.

Compilation flags:

```
static
```

Template:

```
duration_string(Duration,String)
```

Mode and number of proofs:

```
duration_string(++compound,?atom) - zero_or_one
duration_string(?compound,+atom) - zero_or_one
```

Remarks:

- Duration string: An ISO 8601 duration string encodes a length of time (not a calendar date), starting with `P` (or `-P` for negative durations) and optionally using date and time parts such as `P3D` or `P1Y2M3DT4H5M6S`.

Examples:

Parse a duration string

```
duration_string(Duration,'PT45M')
Duration=duration(0,0,0,0,45,0)
```

Format a duration term

```
duration_string(duration(1,2,3,4,5,6),String)
String='P1Y2M3DT4H5M6S'
```

Format a negative duration term

```
duration_string(duration(0,0,-3,0,0,0),String)
String='-P3D'
```

interval_string/2

Conversion between an ISO 8601 interval string and an `interval(Start,End)`, `interval(Start,Duration)`, or `interval(Duration,End)` term where endpoint terms are dates (`[Year,Month,Day]`) or date-times (`date_time/6-7` terms).

Compilation flags:

`static`

Template:

`interval_string(Interval,String)`

Mode and number of proofs:

`interval_string(++compound,?atom)` - `zero_or_one`

`interval_string(?compound,+atom)` - `zero_or_one`

Remarks:

- Interval string: An ISO 8601 interval string encodes a time interval using two parts separated by `/`; each part is a date, a date-time, or a duration, e.g. `2026-02-25/2026-03-01`, `2026-04-07T14:30:00Z/2026-04-07T15:00:00Z`, or `2026-02-25/P3D`.

Examples:

Parse a start/date + duration interval

```
interval_string(Interval,'2026-02-25/P3D')
Interval=interval([2026,2,25],duration(0,0,3,0,0,0))
```

Format a date/date interval

```
interval_string(interval([2026,2,25],[2026,3,1]),String)
String='2026-02-25/2026-03-01'
```

Format a date-time/date-time interval

```
interval_string(interval(date_time(2026,4,7,14,30,0,0),date_time(2026,4,7,
15,0,0,0)),String)
String='2026-04-07T14:30:00Z/2026-04-07T15:00:00Z'
```

Format with a negative duration term

```
interval_string(interval([2026,2,25],duration(0,0,-3,0,0,0)),String)
String='2026-02-25/-P3D'
```

valid_date/3

Validate a given date in the Gregorian calendar.

Compilation flags:

`static`

Template:

`valid_date(Year,Month,Day)`

Mode and number of proofs:

`valid_date(+integer,+integer,+integer)` - `zero_or_one`

Examples:

Yes, the recent millennium was a leap year

```
valid_date(2000,2,29)
yes
```

2004 was also a leap year

```
valid_date(2004,2,29)
yes
```

Only 30 days in April

```
valid_date(2004,4,31)
no
```

1 BC was a leap year

```
valid_date(-1,2,29)
yes
```

`leap_year/1`

Succeed if given year is a leap year in the Gregorian calendar.

Compilation flags:

`static`

Template:

`leap_year(Year)`

`Year` - The Gregorian calendar year to investigate. If free, it binds to the system year.

Mode and number of proofs:

`leap_year(?integer)` - `zero_or_one`

Examples:

No, the prior centenary was not a leap year

```
leap_year(1900)
no
```

The recent millennium

```
leap_year(2000)
yes
```

This year

```
leap_year(Year)
Year=2004
```

This year (equivalent to prior query)

```
leap_year(_)
yes
```

Next centennial

```
leap_year(2100)
no
```

Year 0, equivalent to 1 BC

```
leap_year(0)
yes
```

1 BC

```
leap_year(-1)
yes
```

4 BC

```
leap_year(-4)
no
```

5 BC

```
leap_year(-5)
yes
```

---

calendar_month/3

Compute a calendar month.

Compilation flags:

`static`

Template:

`calendar_month(Year,Month,Calendar)`

`Year` - The calendar year.

`Month` - The calendar month.

`Calendar` - A compound term, `m/3`, composed of three main arguments specifying year, month, and a list of week and week day numbers (calendar body).

Mode and number of proofs:

`calendar_month(?integer,?integer,-compound)` - `zero_or_one`

Examples:

Compute the calendar of March, 2005

```
calendar_month(2005,3,Calendar)
Calendar=m(2005,3,[w(9,[0,1,2,3,4,5,6]),w(10,[7,8,9,10,11,12,13]),w(11,[14,
15,16,17,18,19,20]),w(12,[21,22,23,24,25,26,27]),w(13,[28,29,30,31,0,0,0]),
w(0,[0,0,0,0,0,0,0])])
```

---

easter_day/3

Compute a Gregorian Easter Sunday.

Compilation flags:

`static`

Template:

`easter_day(Year,Month,Day)`

`Year` - Integer specifying the year to be investigated.

`Month` - Month in which Easter Sunday falls for given year.
`Day` - Day of month in which Easter Sunday falls for given year.

Mode and number of proofs:

`easter_day(?integer,-integer,-integer)` - `zero_or_one`

Examples:

Compute Easter Sunday for a particular year

```
easter_day(2006,Month,Day)
Month=4,Day=16
```

Compute Easter Sunday for the current year

```
easter_day(Year,Month,Day)
Year=2005,Month=3,Day=27
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.156 iso_13616

object

1.156.1 `iso_13616`

ISO 13616 IBAN parser and normalizer based on the public ISO and SWIFT registry structure definitions and MOD-97 checksum algorithm.

Availability:

`logtalk_load(iso_13616(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-04

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *iso_13616_protocol*

Uses:

iso_13616_registry

iso_3166
list

Remarks:
(none)

Inherited public predicates:
canonical_iban/2 *formatted_iban/2* *iban/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.156.2 `iso_13616_protocol`

Protocol for ISO 13616 IBAN structure parsing, checksum validation, and normalization.

Availability:
`logtalk_load(iso_13616(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *iban/4*
  - *canonical_iban/2*
  - *formatted_iban/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

iban/4

Parses and validates an ISO 13616 IBAN atom into ISO 3166-1 alpha-2 country code, two-character check digits, and BBAN atom. Input may use lowercase letters and embedded spaces; successful results are normalized to uppercase compact components.

Compilation flags:
`static`

Template:
`iban(IBAN,CountryAlpha2,CheckDigits,BBAN)`
Mode and number of proofs:
`iban(+atom,?atom,?atom,?atom)` - `zero_or_one`

---

canonical_iban/2

Returns the canonical electronic-format IBAN for a valid input by removing spaces and converting letters to uppercase.

Compilation flags:
`static`

Template:
`canonical_iban(IBAN,Canonical)`
Mode and number of proofs:
`canonical_iban(+atom,?atom)` - `zero_or_one`

---

formatted_iban/2

Returns a valid IBAN formatted in groups of four characters separated by spaces.

Compilation flags:
`static`

Template:
`formatted_iban(IBAN,Formatted)`
Mode and number of proofs:
`formatted_iban(+atom,?atom)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.156.3 iso_13616_registry

Public SWIFT IBAN registry snapshot facts for ISO 13616 country-specific validation using a derived Prolog BBAN segment representation.

Availability:
`logtalk_load(iso_13616(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *country_spec/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

country_spec/3

Maps an IBAN country code to the registered total IBAN length and a derived BBAN structure term. BBANPattern is a list of `Type-Count` pairs, where Type is `a` for uppercase letters, `n` for digits, and `c` for uppercase alphanumeric characters. For example, `[a-4, n-6, n-8]` denotes four letters followed by six digits and then eight digits.

Compilation flags:
`static`

Template:
`country_spec(CountryAlpha2,IBANLength,BBANPattern)`
Mode and number of proofs:
`country_spec(?atom,?integer,?list)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.157 iso_3166

object

1.157.1 iso_3166

Facade object for ISO 3166 code queries.

Availability:
`logtalk_load(iso_3166(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *iso_3166_protocol*
Uses:
iso_3166_1
iso_3166_2

Remarks:
(none)

Inherited public predicates:
country/4 *subdivision/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.157.2 `iso_3166_1`

Generated ISO 3166-1 country facts extracted from the UN M49 overview page, which republishes ISO alpha-2 and alpha-3 codes together with M49 numeric codes.

Availability:
`logtalk_load(iso_3166(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Source URL: https://unstats.un.org/unsd/methodology/m49/overview/
- Generated entries: 248.

Inherited public predicates:
(none)

- *Public predicates*
  - *country/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

country/4

Generated ISO 3166-1 country fact table.

Compilation flags:
`static`

Template:
`country(Alpha2,Alpha3,Numeric,Name)`
Mode and number of proofs:
`country(?atom,?atom,?integer,?atom)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.157.3 `iso_3166_2`

Generated ISO 3166-2 subdivision facts extracted from the Debian iso-codes machine-readable JSON snapshot.

Availability:
`logtalk_load(iso_3166(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Source URL: https://salsa.debian.org/iso-codes-team/iso-codes/-/raw/main/data/iso_3166-2.json
- Generated entries: 5046.

Inherited public predicates:
(none)

- *Public predicates*
  - *subdivision/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`subdivision/4`

Generated ISO 3166-2 subdivision fact table.

Compilation flags:
`static`

Template:
`subdivision(Code,CountryAlpha2,Name,Category)`
Mode and number of proofs:
`subdivision(?atom,?atom,?atom,?atom)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.157.4 `iso_3166_protocol`

Protocol for ISO 3166 country and subdivision registry lookups.

Availability:
`logtalk_load(iso_3166(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *country/4*
  - *subdivision/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

country/4

Enumerates ISO 3166-1 country entries as alpha-2, alpha-3, numeric, and English short name tuples.

Compilation flags:
```
static
```

Template:
```
country(Alpha2,Alpha3,Numeric,Name)
```
Mode and number of proofs:
```
country(?atom,?atom,?integer,?atom) - zero_or_more
```

---

subdivision/4

Enumerates ISO 3166-2 subdivision entries as code, parent alpha-2 country code, name, and category tuples.

Compilation flags:
```
static
```

Template:
```
subdivision(Code,CountryAlpha2,Name,Category)
```
Mode and number of proofs:
```
subdivision(?atom,?atom,?atom,?atom) - zero_or_more
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.158 iso_4217

object

1.158.1 `iso_4217`

Generated ISO 4217 active currency and fund facts extracted from the SIX list-one.xml snapshot.

Availability:
`logtalk_load(iso_4217(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *iso_4217_protocol*

Remarks:

- Source publication: 2026-01-01.
- Generated currency entries: 269.
- Generated fund entries: 8.

Inherited public predicates:
(none)

- *Public predicates*
  - *currency/5*
  - *fund_currency/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

currency/5

Generated ISO 4217 active non-fund currency fact table.

Compilation flags:
`static`

Template:
`currency(Alphabetic,Numeric,MinorUnit,Name,Entity)`
Mode and number of proofs:
`currency(?atom,?integer,?term,?atom,?atom)` - `zero_or_more`

---

fund_currency/5

Generated ISO 4217 active fund currency fact table.

Compilation flags:
`static`

Template:
`fund_currency(Alphabetic,Numeric,MinorUnit,Name,Entity)`
Mode and number of proofs:
`fund_currency(?atom,?integer,?term,?atom,?atom)` - `zero_or_more`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.158.2 iso_4217_protocol

Protocol for ISO 4217 active currency and fund code lookups.

Availability:
`logtalk_load(iso_4217(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *currency/5*
  - *fund_currency/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

currency/5

Enumerates ISO 4217 active non-fund entries as alphabetic code, numeric code, minor unit, currency name, and entity tuples. The minor unit is either an integer or the atom `na` when the standard lists `N.A.`.

Compilation flags:
`static`

Template:
`currency(Alphabetic,Numeric,MinorUnit,Name,Entity)`
Mode and number of proofs:
`currency(?atom,?integer,?term,?atom,?atom)` - `zero_or_more`

fund_currency/5

Enumerates ISO 4217 active fund entries as alphabetic code, numeric code, minor unit, currency name, and entity tuples. The minor unit is either an integer or the atom `na` when the standard lists `N.A.`.

Compilation flags:
`static`

Template:
`fund_currency(Alphabetic,Numeric,MinorUnit,Name,Entity)`
Mode and number of proofs:
`fund_currency(?atom,?integer,?term,?atom,?atom)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.159 iso_639

object

1.159.1 iso_639

Facade object for ISO 639 language and language-group queries.

Availability:
`logtalk_load(iso_639(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *iso_639_protocol*
Uses:
iso_639_1
iso_639_2

iso_639_3
iso_639_5

Remarks:
(none)

Inherited public predicates:
language/3 *language/5* *language_code/5* *language_group/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.159.2 `iso_639_1`

Generated ISO 639-1 fact table.

Availability:
`logtalk_load(iso_639(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Source update: 2026-01-15.
- Generated entries: 183.
- Source: Library of Congress ISO 639-2 code list.

Inherited public predicates:
(none)

- *Public predicates*
  - `language/3`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`language/3`

Generated ISO 639-1 fact table.

Compilation flags:
`static`

Template:
`language(Alpha2,Alpha3,Name)`
Mode and number of proofs:
`language(?atom,?atom,?atom)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.159.3 iso_639_2

Generated ISO 639-2 fact table.

Availability:
`logtalk_load(iso_639(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Source update: 2026-01-15.
- Generated entries: 487.
- Source: Library of Congress ISO 639-2 code list.

Inherited public predicates:
(none)

- *Public predicates*
  - *language_code/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

language_code/5

Generated ISO 639-2 fact table.

Compilation flags:
`static`

Template:
`language_code(Bibliographic,Terminologic,Alpha2,Name,Class)`
Mode and number of proofs:
`language_code(?atom,?atom,?atom,?atom,?atom) - zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.159.4 `iso_639_3`

Generated ISO 639-3 fact table.

Availability:
`logtalk_load(iso_639(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Source update: 2026-04-15.
- Generated entries: 7929.
- Source: SIL ISO 639-3 tab-delimited code set.

Inherited public predicates:
(none)

- *Public predicates*
  - *language/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`language/5`

Generated ISO 639-3 fact table.

Compilation flags:
`static`

Template:
`language(Alpha3,Alpha2,Scope,Type,Name)`
Mode and number of proofs:
`language(?atom,?atom,?atom,?atom,?atom)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.159.5 `iso_639_5`

Generated ISO 639-5 fact table.

Availability:
`logtalk_load(iso_639(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Source update: 2026-04-15.
- Generated entries: 115.
- Source: Library of Congress ISO 639-5 identifier list.

Inherited public predicates:
(none)

- *Public predicates*
  - *language_group/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

language_group/2

Generated ISO 639-5 fact table.

Compilation flags:
`static`

Template:
`language_group(Code,Name)`
Mode and number of proofs:
`language_group(?atom,?atom)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.159.6 iso_639_protocol

Protocol for ISO 639 language and language-group lookups.

Availability:
`logtalk_load(iso_639(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *language/3*
  - *language_code/5*
  - *language/5*
  - *language_group/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

language/3

Enumerates ISO 639-1 entries as alpha-2, alpha-3, and English name tuples.

Compilation flags:
`static`

Template:
`language(Alpha2,Alpha3,Name)`

Mode and number of proofs:
`language(?atom,?atom,?atom)` - `zero_or_more`

---

language_code/5

Enumerates ISO 639-2 entries as bibliographic code, terminologic code, alpha-2 code, English name, and class tuples. The class is one of `individual`, `macrolanguage`, `collective`, `special`, or `local_use`.

Compilation flags:
`static`

Template:
`language_code(Bibliographic,Terminologic,Alpha2,Name,Class)`

Mode and number of proofs:

`language_code(?atom,?atom,?atom,?atom,?atom)` - `zero_or_more`

`language/5`

Enumerates ISO 639-3 entries as alpha-3 code, alpha-2 code, scope, type, and English name tuples. The scope is one of `individual`, `macrolanguage`, `collective`, or `special` and the type is one of `living`, `extinct`, `ancient`, `constructed`, or `special`.

Compilation flags:

`static`

Template:

`language(Alpha3,Alpha2,Scope,Type,Name)`

Mode and number of proofs:

`language(?atom,?atom,?atom,?atom,?atom)` - `zero_or_more`

`language_group/2`

Enumerates ISO 639-5 entries as group code and English name tuples.

Compilation flags:

`static`

Template:

`language_group(Code,Name)`

Mode and number of proofs:

`language_group(?atom,?atom)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.160 iso_9362

object

1.160.1 `iso_9362`

ISO 9362 BIC parser and normalizer based on the public Swift and ISO 9362 structure definitions.

Availability:
`logtalk_load(iso_9362(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *iso_9362_protocol*
Uses:
iso_3166

Remarks:
(none)

Inherited public predicates:
bic/5 canonical_bic/2 primary_office_bic/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.160.2 `iso_9362_protocol`

Protocol for ISO 9362 BIC structure parsing and normalization.

Availability:
`logtalk_load(iso_9362(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *bic/5*
  - *canonical_bic/2*
  - *primary_office_bic/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

bic/5

Parses and validates an ISO 9362 BIC atom into business party prefix, ISO 3166-1 alpha-2 country code, business party suffix, and branch identifier. Eight-character BICs are normalized to the primary-office branch code XXX.

Compilation flags:
static

Template:
bic(BIC,Prefix,CountryAlpha2,Suffix,Branch)
Mode and number of proofs:
bic(+atom,?atom,?atom,?atom,?atom) - zero_or_one

---

canonical_bic/2

Returns the canonical 11-character form of a valid ISO 9362 BIC, using the branch code XXX when the input is an eight-character primary-office BIC.

Compilation flags:
static

Template:
canonical_bic(BIC,Canonical)
Mode and number of proofs:
canonical_bic(+atom,?atom) - zero_or_one

---

primary_office_bic/1

True when a valid ISO 9362 BIC denotes the primary office, either because it is eight characters long or because its branch identifier is XXX.

Compilation flags:
static

Template:
primary_office_bic(BIC)
Mode and number of proofs:
primary_office_bic(+atom) - zero_or_one

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.161 isolation_forest_anomaly_detector

object

1.161.1 `isolation_forest_anomaly_detector`

Extended Isolation Forest (EIF) algorithm for anomaly detection. Implements the improved version described by Hariri et al. (2019) that uses random hyperplane cuts instead of axis-aligned cuts, eliminating score bias artifacts. Builds an ensemble of isolation trees from baseline training examples selected from a dataset object implementing the `anomaly_dataset_protocol` protocol. Missing attribute values are represented using anonymous variables.

Availability:
`logtalk_load(isolation_forest_anomaly_detector(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2026-10-10

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *anomaly_detector_common*

Uses:
fast_random(Algorithm)
format
integer
list
numberlist
pairs
type

Remarks:
(none)

Inherited public predicates:
anomaly_detector_options/2 *check_anomaly_detector/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict/4* *print_anomaly_detector/1* *valid_anomaly_detector/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *score/3*
  - *score_all/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

score/3

Computes the anomaly score for a given instance using the learned model. The instance is a list of `Attribute-Value` pairs where missing values are represented using anonymous variables. The score is in the range `[0.0, 1.0]`. Scores close to `1.0` indicate anomalies. Scores close to `0.5` or below indicate normal instances.

Compilation flags:

```
static
```

Template:

```
score(Model,Instance,Score)
```

Mode and number of proofs:

```
score(+compound,+list,-float) - one
```

---

score_all/3

Computes the anomaly scores for all instances in the dataset. Returns a list of `Id-Class-Score` triples sorted by descending anomaly score.

Compilation flags:

```
static
```

Template:

```
score_all(Dataset,Model,Scores)
```

Mode and number of proofs:

```
score_all(+object_identifier,+compound,-list) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

anomaly_dataset_protocol, *anomaly_detector_protocol*, *knn_distance_anomaly_detector*, *lof_anomaly_detector*

1.162 issue_creator

object

1.162.1 `issue_creator`

Support for automatically creating bug report issues for failed tests in GitHub, GitLab, Azure DevOps, Gitea, Jira, or Linear servers.

Availability:
`logtalk_load(issue_creator(loader))`

Author: Paulo Moura
Version: 0:14:1
Date: 2026-11-17

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
base64
git
list
os
term_io
user
utf_16le

Remarks:

- Usage: This tool is automatically loaded and used from the `logtalk_tester` automation script when using its `-b` option. See the script man page for details.

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *command_text/4*
  - *title/3*
  - *search_arguments/5*
  - *search_command/4*
  - *empty_search/2*
  - *jira_search_command/3*
  - *jira_list_command/4*
  - *jira_jql/3*
  - *jira_label_conditions/2*
  - *azure_list_command/4*
  - *azure_search_command/6*
  - *azure_states_query/1*
  - *azure_wiql/3*
  - *powershell_command/2*
  - *valid_labels/1*
  - *valid_label_code/1*
  - *command_arguments/3*
  - *run_cli/3*
  - *null_redirect/2*
  - *create_arguments/7*
  - *option_arguments/3*
  - *label_arguments/2*
  - *repeated_labels/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

command_text/4

Renders an executable and literal arguments for the given operating system.

Compilation flags:

```
static
```

Template:

```
command_text(OperatingSystem,Executable,Arguments,Command)
```
Mode and number of proofs:
```
command_text(+atom,+atom,+list(atom),-atom) - zero_or_one
```

---

title/3

Constructs a stable title identifying the test and test set directory.

Compilation flags:
```
static
```

Template:
```
title(Test,Directory,Title)
```
Mode and number of proofs:
```
title(+term,+atom,-atom) - one
```

---

search_arguments/5

Constructs a title and label search requesting at most one matching issue.

Compilation flags:
```
static
```

Template:
```
search_arguments(Server,Title,Labels,Executable,Arguments)
```
Mode and number of proofs:
```
search_arguments(+atom,+atom,+atom,-atom,-list(atom)) - zero_or_one
```

---

search_command/4

Constructs a command that succeeds only when the query succeeds and reports no matching issue.

Compilation flags:
```
static
```

Template:
```
search_command(OperatingSystem,Server,Query,Command)
```
Mode and number of proofs:
```
search_command(+atom,+atom,+atom,-atom) - one
```

---

empty_search/2

Returns the CLI output for a successful search with no matching issue.

Compilation flags:
`static`

Template:
`empty_search(Server,Output)`
Mode and number of proofs:
`empty_search(+atom,-atom)` - `one`

---

jira_search_command/3

Constructs a command accepting only the Jira CLI no-result diagnostic after an unsuccessful search.

Compilation flags:
`static`

Template:
`jira_search_command(OperatingSystem,Query,Command)`
Mode and number of proofs:
`jira_search_command(+atom,+atom,-atom)` - `one`

---

jira_list_command/4

Constructs a command searching for at most one matching Jira bug.

Compilation flags:
`static`

Template:
`jira_list_command(OperatingSystem,Title,Labels,Command)`
Mode and number of proofs:
`jira_list_command(+atom,+atom,+atom,-atom)` - `zero_or_one`

---

jira_jql/3

Constructs a JQL phrase search with an escaped title and validated labels.

Compilation flags:
```
static
```

Template:
```
jira_jql(Title,Labels,Query)
```
Mode and number of proofs:
```
jira_jql(+atom,+atom,-atom) - one
```

---

jira_label_conditions/2

Constructs a conjunction requiring every Jira label.

Compilation flags:
```
static
```

Template:
```
jira_label_conditions(Labels,Conditions)
```
Mode and number of proofs:
```
jira_label_conditions(+list(atom),-atom) - one
```

---

azure_list_command/4

Constructs an Azure duplicate check using the configured project and its Bug workflow states.

Compilation flags:
```
static
```

Template:
```
azure_list_command(OperatingSystem,Title,Labels,Command)
```
Mode and number of proofs:
```
azure_list_command(+atom,+atom,+atom,-atom) - zero_or_one
```

---

azure_search_command/6

Accepts absence only after successful project, workflow, and issue queries.

Compilation flags:

```
static
```

Template:

```
azure_search_command(Shell,ProjectQuery,StatesQuery,IssueQuery,QuotedWIQL,
Program)
```

Mode and number of proofs:

```
azure_search_command(+atom,+atom,+atom,+atom,+atom,-atom) - one
```

---

azure_states_query/1

Returns a native Azure CLI query selecting nonterminal Bug state names as double-quoted WIQL literals.

Compilation flags:

```
static
```

Template:

```
azure_states_query(Query)
```

Mode and number of proofs:

```
azure_states_query(-atom) - one
```

---

azure_wiql/3

Constructs the title and tag portion of an Azure Bug query; project and workflow constraints are appended at execution.

Compilation flags:

```
static
```

Template:

```
azure_wiql(Title,Labels,Query)
```

Mode and number of proofs:

```
azure_wiql(+atom,+atom,-atom) - one
```

---

powershell_command/2

Encodes a PowerShell program to avoid cmd.exe expansion of query data.

Compilation flags:

```
static
```

Template:

```
powershell_command(Program,Command)
```

Mode and number of proofs:

```
powershell_command(+atom,-atom) - one
```

---

valid_labels/1

Checks for comma-separated nonempty ASCII labels starting with a letter or digit and containing only letters, digits, underscores, or hyphens.

Compilation flags:

```
static
```

Template:

```
valid_labels(Labels)
```

Mode and number of proofs:

```
valid_labels(+atom) - zero_or_one
```

---

valid_label_code/1

Checks for an ASCII letter, digit, underscore, or hyphen.

Compilation flags:

```
static
```

Template:

```
valid_label_code(Code)
```

Mode and number of proofs:

```
valid_label_code(+character_code) - zero_or_one
```

---

command_arguments/3

Renders individually quoted command arguments.

Compilation flags:
`static`

Template:
`command_arguments(Arguments,OperatingSystem,Text)`
Mode and number of proofs:
`command_arguments(+list(atom),+atom,-atom)` - `zero_or_one`

---

run_cli/3

Runs a command, discarding its output and succeeding only on a successful exit status.

Compilation flags:
`static`

Template:
`run_cli(OperatingSystem,Executable,Arguments)`
Mode and number of proofs:
`run_cli(+atom,+atom,+list(atom))` - `zero_or_one_or_error`

Exceptions:
An operating-system operation raises an error:
`error(system_error,Context)`

---

null_redirect/2

Returns the operating-system redirection for discarding standard output and standard error.

Compilation flags:
`static`

Template:
`null_redirect(OperatingSystem,Redirect)`
Mode and number of proofs:
`null_redirect(+atom,-atom)` - `one`

---

create_arguments/7

Constructs literal issue creation arguments for a tracker.

Compilation flags:
```
static
```

Template:
```
create_arguments(Server,Title,Description,Labels,Assignee,Executable,Arguments)
```
Mode and number of proofs:
```
create_arguments(+atom,+atom,+atom,+atom,+atom,-atom,-list(atom)) - one
```

---

option_arguments/3

Omits an option when its value is empty.

Compilation flags:
```
static
```

Template:
```
option_arguments(Option,Value,Arguments)
```
Mode and number of proofs:
```
option_arguments(+atom,+atom,-list(atom)) - one
```

---

label_arguments/2

Constructs repeated label arguments from comma-separated labels.

Compilation flags:
```
static
```

Template:
```
label_arguments(Labels,Arguments)
```
Mode and number of proofs:
```
label_arguments(+atom,-list(atom)) - one
```

---

repeated_labels/2

Pairs each label with its command option.

Compilation flags:
`static`

Template:
`repeated_labels(Labels,Arguments)`
Mode and number of proofs:
`repeated_labels(+list(atom),-list(atom))-one`

Operators

(none)

1.163 jaccard_recommender

object

1.163.1 jaccard_recommender

Content-based recommender using binary item features, support-filtered positive-feedback profiles, and Jaccard relevance scores.

Availability:
`logtalk_load(jaccard_recommender(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *recommender_common*
`public` *item_content_dataset_validation*
Uses:
jaccard_similarity
list
type

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_recommender/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_recommender/1* *recommend/4* *recommender_options/2* *score/4* *valid_option/1* *valid_options/1* *valid_recommender/1*

- *Public predicates*
  - `score_all/4`
  - `score_content/4`
  - `extend_catalog/3`
  - `replace_content/3`
  - `update_ratings/3`
  - `remove_ratings/3`
- *Protected predicates*
- *Private predicates*
  - `rating_removal_records/3`
  - `check_rating_updates/3`
  - `check_catalog_replacements/2`
  - `check_catalog_additions/2`
  - `catalog_score/5`
  - `score_catalog_items/5`
  - `binary_contents/2`
  - `binary_descriptor/2`
  - `binary_pairs/2`
  - `check_finite_ratings/1`
- *Operators*

Public predicates

`score_all/4`

Scores catalog items in input order, preserving duplicates and validating the model once. An empty list returns an empty list after validating the model and user.

Compilation flags:

`static`

Template:

`score_all(Recommender,User,Items,Scores)`

Mode and number of proofs:

`score_all(+compound,+atomic,+list(atomic),-list(pair))` - `one_or_error`

Exceptions:

A required argument or item identifier is a variable:

`instantiation_error`

The model is invalid:

`domain_error(recommender,Recommender)`

User or an item identifier is not atomic:

`type_error(atomic,Identifier)`

Items is not a proper list:

```
type_error(list,Items)
```

An item is absent from the learned catalog:

```
domain_error(catalog_item,Item)
```

---

score_content/4

Scores supplied binary content against a learned user profile without changing the model or catalog. Either descriptor kind is accepted regardless of the training representation.

Compilation flags:

```
static
```

Template:

```
score_content(Recommender,User,Content,Score)
```

Mode and number of proofs:

```
score_content(+compound,+atomic,+compound,-float) - one_or_error
```

Exceptions:

A required argument, feature, entry, or weight is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

User is not atomic:

```
type_error(atomic,User)
```

The content descriptor is invalid:

```
domain_error(item_content,Content)
```

A content list is not a proper list:

```
type_error(list,List)
```

A vector entry is not a pair:

```
type_error(pair,Entry)
```

A vector repeats a feature key:

```
domain_error(duplicate_feature,Feature)
```

A vector weight is not numeric:

```
type_error(number,Weight)
```

A vector weight is negative or nonfinite:

```
domain_error(non_negative_finite_weight,Weight)
```

A vector weight is neither zero nor one:

```
domain_error(binary_weight,Weight)
```

---

extend_catalog/3

Returns a model with additional unrated items of the existing content representation, preserving ratings and profiles. An empty extension returns the validated original model.

Compilation flags:

```
static
```

Template:

```
extend_catalog(Recommender,ItemContents,UpdatedRecommender)
```

Mode and number of proofs:

```
extend_catalog(+compound,+list(pair),-compound) - one_or_error
```

Exceptions:

A required argument, entry, identifier, feature, or weight is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

A supplied list is not a proper list:

```
type_error(list,List)
```

A content declaration or vector entry is not a pair:

```
type_error(pair,Entry)
```

An item identifier is not atomic:

```
type_error(atomic,Item)
```

An identifier already belongs to the catalog:

```
domain_error(new_catalog_item,Item)
```

An identifier occurs more than once in the extension:

```
domain_error(duplicate_item,Item)
```

A content descriptor is invalid:

```
domain_error(item_content,Content)
```

Content representations are mixed or differ from the catalog:

```
domain_error(content_representation,Content)
```

A vector repeats a feature key:

```
domain_error(duplicate_feature,Feature)
```

A vector weight is not numeric:

```
type_error(number,Weight)
```

A vector weight is negative or nonfinite:

```
domain_error(non_negative_finite_weight,Weight)
```

A vector weight is neither zero nor one:

```
domain_error(binary_weight,Weight)
```

replace_content/3

Returns a model with replaced catalog content and rebuilt profiles. An empty replacement returns the validated original model.

Compilation flags:

```
static
```

Template:

```
replace_content(Recommender,ItemContents,UpdatedRecommender)
```

Mode and number of proofs:

```
replace_content(+compound,+list(pair),-compound) - one_or_error
```

Exceptions:

A required argument, entry, identifier, feature, or weight is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

A supplied list is not a proper list:

```
type_error(list,List)
```

A content declaration or vector entry is not a pair:

```
type_error(pair,Entry)
```

An item identifier is not atomic:

```
type_error(atomic,Item)
```

An item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

An identifier occurs more than once in the replacements:

```
domain_error(duplicate_item,Item)
```

A content descriptor is invalid:

```
domain_error(item_content,Content)
```

Content representations are mixed or differ from the catalog:

```
domain_error(content_representation,Content)
```

A vector repeats a feature key:

```
domain_error(duplicate_feature,Feature)
```

A vector weight is not numeric:

```
type_error(number,Weight)
```

A vector weight is negative or nonfinite:

```
domain_error(non_negative_finite_weight,Weight)
```

A vector weight is neither zero nor one:

```
domain_error(binary_weight,Weight)
```

update_ratings/3

Returns a model with inserted or replaced user-item ratings and rebuilt profiles, accepting new users but requiring catalog items. An empty update returns the validated original model.

Compilation flags:

```
static
```

Template:

```
update_ratings(Recommender,Ratings,UpdatedRecommender)
```

Mode and number of proofs:

```
update_ratings(+compound,+list(compound),-compound) - one_or_error
```

Exceptions:

A required argument, record, identifier, or value is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

Ratings is not a proper list:

```
type_error(list,Ratings)
```

An entry is not a rating record:

```
type_error(rating,Entry)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

A rating value is not numeric:

```
type_error(number,Rating)
```

A rating value is nonfinite:

```
domain_error(finite_rating,Rating)
```

A rating value is outside the stored scale:

```
domain_error(rating_scale(Min,Max),Rating)
```

An item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

A user-item pair occurs more than once in the updates:

```
domain_error(duplicate_rating,User-Item)
```

---

remove_ratings/3

Returns a model with requested user-item ratings removed and profiles rebuilt. Missing and repeated pairs are accepted; no matching ratings returns the validated original model.

Compilation flags:

```
static
```

Template:

```
remove_ratings(Recommender,UserItemPairs,UpdatedRecommender)
```

Mode and number of proofs:

```
remove_ratings(+compound,+list(pair),-compound) - one_or_error
```

Exceptions:

A required argument, entry, or identifier is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

UserItemPairs is not a proper list:

```
type_error(list,UserItemPairs)
```

An entry is not a pair:

```
type_error(pair,Entry)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

No ratings would remain:

```
domain_error(non_empty_ratings,[])
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

rating_removal_records/3

Validates user-item pairs and collects matching stored ratings, ignoring missing pairs.

Compilation flags:

```
static
```

Template:

```
rating_removal_records(UserItemPairs,Ratings,Removals)
```

Mode and number of proofs:

```
rating_removal_records(+list(pair),+list(compound),-list(compound)) - one_or_error
```

Exceptions:

An entry or identifier is a variable:

```
instantiation_error
```

An entry is not a pair:

```
type_error(pair,Entry)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

---

check_rating_updates/3

Checks rating records against the catalog and stored rating scale.

Compilation flags:

```
static
```

Template:

```
check_rating_updates(Ratings,Vectors,Scale)
```

Mode and number of proofs:

```
check_rating_updates(+list(compound),+list(pair),+term) - one_or_error
```

Exceptions:

A record, identifier, or value is a variable:

```
instantiation_error
```

An entry is not a rating record:

```
type_error(rating,Entry)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

A rating value is not numeric:

```
type_error(number,Rating)
```

A rating value is nonfinite:

```
domain_error(finite_rating,Rating)
```

A rating value is outside the stored scale:

```
domain_error(rating_scale(Min,Max),Rating)
```

An item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

---

check_catalog_replacements/2

Checks that replacement declarations are pairs with atomic catalog identifiers.

Compilation flags:

```
static
```

Template:

```
check_catalog_replacements(ItemContents,Vectors)
```

Mode and number of proofs:

```
check_catalog_replacements(+list(pair),+list(pair)) - one_or_error
```

Exceptions:

An entry or identifier is a variable:

```
instantiation_error
```

An entry is not a pair:

```
type_error(pair,Entry)
```

An identifier is not atomic:

```
type_error(atomic,Item)
```

An identifier is absent from the catalog:
`domain_error(catalog_item,Item)`

---

check_catalog_additions/2

Checks that content declarations are pairs with atomic identifiers absent from the existing catalog.

Compilation flags:
`static`

Template:
`check_catalog_additions(ItemContents,Vectors)`
Mode and number of proofs:
`check_catalog_additions(+list(pair),+list(pair))` - `one_or_error`

Exceptions:
An entry or identifier is a variable:
`instantiation_error`
An entry is not a pair:
`type_error(pair,Entry)`
An identifier is not atomic:
`type_error(atomic,Item)`
An identifier already belongs to the catalog:
`domain_error(new_catalog_item,Item)`

---

catalog_score/5

Scores an instantiated catalog identifier using validated model data.

Compilation flags:
`static`

Template:
`catalog_score(Vectors,Profiles,User,Item,Score)`
Mode and number of proofs:
`catalog_score(+list(pair),+list(pair),+atomic,+atomic,-float)` - `one_or_error`

Exceptions:
The item is absent from the learned catalog:
`domain_error(catalog_item,Item)`

---

score_catalog_items/5

Validates and scores catalog identifiers in order using validated model data.

Compilation flags:
`static`

Template:
`score_catalog_items(Vectors,Profiles,User,Items,Scores)`
Mode and number of proofs:
`score_catalog_items(+list(pair),+list(pair),+atomic,+list(atomic),-list(pair))` - `one_or_error`

Exceptions:
An item identifier is a variable:
`instantiation_error`
An item identifier is not atomic:
`type_error(atomic,Item)`
An item is absent from the learned catalog:
`domain_error(catalog_item,Item)`

---

binary_contents/2

Converts canonical descriptors into sorted binary feature vectors, ignoring feature multiplicity.

Compilation flags:
`static`

Template:
`binary_contents(Contents,Vectors)`
Mode and number of proofs:
`binary_contents(+list(pair),-list(pair))` - `one_or_error`

Exceptions:
A nonzero canonical vector weight is not one:
`domain_error(binary_weight,Weight)`

---

binary_descriptor/2

Converts a canonical descriptor to a binary feature vector.

Compilation flags:
`static`

Template:

```
binary_descriptor(Content,Vector)
```

Mode and number of proofs:

```
binary_descriptor(+compound,-list(pair)) - one_or_error
```

Exceptions:

A nonzero canonical vector weight is not one:

```
domain_error(binary_weight,Weight)
```

---

binary_pairs/2

Checks that nonzero canonical weights are one and replaces them with integer unit weights.

Compilation flags:

```
static
```

Template:

```
binary_pairs(Pairs,Vector)
```

Mode and number of proofs:

```
binary_pairs(+list(pair),-list(pair)) - one_or_error
```

Exceptions:

A nonzero canonical vector weight is not one:

```
domain_error(binary_weight,Weight)
```

---

check_finite_ratings/1

Checks that rating values are finite.

Compilation flags:

```
static
```

Template:

```
check_finite_ratings(Ratings)
```

Mode and number of proofs:

```
check_finite_ratings(+list(compound)) - one_or_error
```

Exceptions:

A rating is not finite:

```
domain_error(finite_rating,Rating)
```

Operators

(none)

See also

recommender_protocol, *item_content_dataset_protocol*, *jaccard_similarity*

1.164 java

object

1.164.1 `java`

Abstract interface to JPL API utility predicates.

Availability:
`logtalk_load(java(loader))`

Author: Paulo Moura
Version: 1:8:0
Date: 2023-03-15

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *java_utils_protocol*
Uses:
user

Remarks:
(none)

Inherited public predicates:
array_list/2 *array_to_list/2* *array_to_terms/2* *array_to_terms/3* *decode_exception/2* *decode_exception/3* *false/1* *is_false/1* *is_null/1* *is_object/1* *is_true/1* *is_void/1* *iterator_element/2* *list_to_array/2* *map_element/2* *null/1* *set_element/2* *terms_to_array/2* *true/1* *value_reference/2* *void/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

java(Reference,ReturnValue), *java(Reference)*, *java_hook*

object

1.164.2 `java(Reference)`

- `Reference` - Either a class name or a Java reference to an object.

Minimal abstraction of the JPL API for calling Java from Logtalk using familiar message-sending syntax and a `forward/1` handler to resolve methods.

Availability:
`logtalk_load(java(loader))`

Author: Paulo Moura and Sergio Castro
Version: 1:0:1
Date: 2019-06-13

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *java(Reference,_)*

Remarks:

- Usage: Send to this object any valid message as listed in the JavaDocs for the given reference.

Inherited public predicates:
forward/1 *get_field/2* *invoke/1* *invoke/2* *new/1* *new/2* *set_field/2*

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

java(Reference,ReturnValue), *java*, *java_hook*

object

1.164.3 `java(Reference,ReturnValue)`

- `Reference` - Either a class name or a Java reference to an object.
- `ReturnValue` - Value returned by a method call (possibly the Java value void).

Minimal abstraction of the JPL API for calling Java from Logtalk using familiar message-sending syntax and a `forward/1` handler to resolve methods.

Availability:
`logtalk_load(java(loader))`

Author: Paulo Moura and Sergio Castro
Version: 1:4:0
Date: 2023-03-13

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *forwarding*
`public` *java_access_protocol*

Remarks:

- Usage: Send to this object any valid message as listed in the JavaDocs for the given reference.

Inherited public predicates:

forward/1 *get_field/2* *invoke/1* *invoke/2* *new/1* *new/2* *set_field/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

java(Reference), *java*, *java_hook*

protocol

1.164.4 `java_access_protocol`

Protocol for a minimal abstraction for calling Java from Logtalk using familiar message-sending syntax.

Availability:
`logtalk_load(java(loader))`

Author: Paulo Moura and Sergio Castro
Version: 1:2:1
Date: 2023-03-16

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *get_field/2*
  - *set_field/2*
  - *new/2*
  - *new/1*
  - *invoke/1*
  - *invoke/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

get_field/2

Gets the value of a class or object field.

Compilation flags:
```
static
```

Template:
```
get_field(Field,Value)
```
Mode and number of proofs:
```
get_field(+atom,?nonvar) - zero_or_one
```

---

set_field/2

Sets the value of a class or object field.

Compilation flags:
```
static
```

Template:
```
set_field(Field,Value)
```
Mode and number of proofs:
```
set_field(+atom,+nonvar) - one
```

---

new/2

Creates a new instance using the specified parameter values.

Compilation flags:
`static`

Template:
`new(Parameters,Instance)`
Mode and number of proofs:
`new(+list(nonvar),-reference)` - `one`

---

new/1

Creates a new instance using default parameter values.

Compilation flags:
`static`

Template:
`new(Instance)`
Mode and number of proofs:
`new(-reference)` - `one`

---

invoke/1

Invokes a method. This is a more efficient compared with relying on the `forward/1` handler to resolve methods.

Compilation flags:
`static`

Template:
`invoke(Method)`
Mode and number of proofs:
`invoke(@nonvar)` - `one`

---

invoke/2

Invokes a method. This is a more efficient compared with relying on the `forward/1` handler to resolve methods.

Compilation flags:
`static`

Template:
`invoke(Functor,Arguments)`
Mode and number of proofs:
`invoke(@nonvar,@list)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.164.5 java_hook

Hook object to optimize messages to the `java/1-2` objects that otherwise would trigger the `forward/1` handler.

Availability:
`logtalk_load(java(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2019-06-13

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Remarks:

- Usage: Compile source files with messages to the `java/1-2` objects using the compiler option `hook(java_hook)`.

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

java(Reference,ReturnValue), *java(Reference)*

protocol

1.164.6 `java_utils_protocol`

Abstract interface to Java utility predicates.

Availability:
`logtalk_load(java(loader))`

Author: Paulo Moura
Version: 1:6:0
Date: 2023-03-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *value_reference/2*
  - *true/1*
  - *false/1*
  - *void/1*
  - *null/1*
  - *is_true/1*
  - *is_false/1*
  - *is_void/1*
  - *is_null/1*
  - *is_object/1*
  - *terms_to_array/2*
  - *array_to_terms/3*
  - *array_to_terms/2*
  - *array_to_list/2*
  - *list_to_array/2*
  - *array_list/2*
  - *iterator_element/2*
  - *map_element/2*
  - *set_element/2*
  - *decode_exception/2*
  - *decode_exception/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

value_reference/2

Returns an opaque term that represents the Java value with the given name.

Compilation flags:

```
static
```

Template:

```
value_reference(Value,Reference)
```

Mode and number of proofs:

```
value_reference(?atom,--ground) - one_or_more
```

true/1

Returns an opaque term that represents the Java value *true*.

Compilation flags:
`static`

Template:
`true(Reference)`
Mode and number of proofs:
`true(--ground)` - `one`

---

false/1

Returns an opaque term that represents the Java value *false*.

Compilation flags:
`static`

Template:
`false(Reference)`
Mode and number of proofs:
`false(--ground)` - `one`

---

void/1

Returns an opaque term that represents the Java value *void*.

Compilation flags:
`static`

Template:
`void(Reference)`
Mode and number of proofs:
`void(--ground)` - `one`

---

null/1

Returns an opaque term that represents the Java value *null*.

Compilation flags:
`static`

Template:
`null(Reference)`
Mode and number of proofs:
`null(--ground)` - `one`

---

is_true/1

True when the argument is the Java value *true*. Fails if the argument is not instantiated.

Compilation flags:
`static`

Template:
`is_true(Reference)`
Mode and number of proofs:
`is_true(@term)` - `zero_or_one`

---

is_false/1

True when the argument is the Java value *false*. Fails if the argument is not instantiated.

Compilation flags:
`static`

Template:
`is_false(Reference)`
Mode and number of proofs:
`is_false(@term)` - `zero_or_one`

---

is_void/1

True when the argument is the Java value *void*. Fails if the argument is not instantiated.

Compilation flags:
`static`

Template:
`is_void(Reference)`
Mode and number of proofs:
`is_void(@term)` - `zero_or_one`

---

is_null/1

True when the argument is the Java value *null*. Fails if the argument is not instantiated.

Compilation flags:
`static`

Template:
`is_null(Reference)`
Mode and number of proofs:
`is_null(@term)` - `zero_or_one`

---

is_object/1

True when the argument is a reference to a Java object. Fails if the argument is not instantiated.

Compilation flags:
`static`

Template:
`is_object(Reference)`
Mode and number of proofs:
`is_object(@term)` - `zero_or_one`

---

terms_to_array/2

Converts a list of ground Prolog terms to an array (a Java reference).

Compilation flags:

```
static
```

Template:

```
terms_to_array(Terms,Array)
```

Mode and number of proofs:

```
terms_to_array(++list(ground),-array) - one
```

---

array_to_terms/3

Converts an array (a Java reference) to a list of ground Prolog terms returning also its length. The array elements must be atoms, integers, floats, or compound terms. Fails otherwise.

Compilation flags:

```
static
```

Template:

```
array_to_terms(Array,Terms,Length)
```

Mode and number of proofs:

```
array_to_terms(+array,-list(ground),-integer) - one
```

---

array_to_terms/2

Converts an array (a Java reference) to a list of ground Prolog terms. The array elements must be atoms, integers, floats, or ground compound terms. Fails otherwise.

Compilation flags:

```
static
```

Template:

```
array_to_terms(Array,Terms)
```

Mode and number of proofs:

```
array_to_terms(+array,-list(term)) - one
```

---

array_to_list/2

Converts an array (a Java reference) to a list of Java references or their values.

Compilation flags:
```
static
```

Template:
```
array_to_list(Array,List)
```
Mode and number of proofs:
```
array_to_list(+array,-list) - one
```

---

list_to_array/2

Converts a list of Java references or values to an array (a Java reference).

Compilation flags:
```
static
```

Template:
```
list_to_array(List,Array)
```
Mode and number of proofs:
```
list_to_array(+list,-array) - one
```

---

array_list/2

Converts between an array (a Java reference) and a list of Java references or their values. Deprecated. Use the `array_to_list/2` and `list_to_array/2` predicates instead.

Compilation flags:
```
static
```

Template:
```
array_list(Array,List)
```
Mode and number of proofs:
```
array_list(+array,-list) - one
array_list(-array,+list) - one
```

---

iterator_element/2

Enumerates, by backtracking, all iterator elements.

Compilation flags:
`static`

Template:
`iterator_element(Iterator,Element)`
Mode and number of proofs:
`iterator_element(+iterator,-element)` - `zero_or_more`

---

map_element/2

Enumerates, by backtracking, all map elements.

Compilation flags:
`static`

Template:
`map_element(Map,Element)`
Mode and number of proofs:
`map_element(+iterator,-element)` - `zero_or_more`

---

set_element/2

Enumerates, by backtracking, all set elements.

Compilation flags:
`static`

Template:
`set_element(Set,Element)`
Mode and number of proofs:
`set_element(+iterator,-element)` - `zero_or_more`

---

decode_exception/2

Decodes an exception into its corresponding cause. Fails if the exception is not a Java exception.

Compilation flags:
```
static
```

Template:
```
decode_exception(Exception,Cause)
```
Mode and number of proofs:
```
decode_exception(+callable,-atom) - zero_or_one
```

---

decode_exception/3

Decodes an exception into its corresponding cause and a stack trace. Fails if the exception is not a Java exception.

Compilation flags:
```
static
```

Template:
```
decode_exception(Exception,Cause,StackTrace)
```
Mode and number of proofs:
```
decode_exception(+callable,-atom,-list(atom)) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.165 json

object

1.165.1 `json`

JSON parser and generator. Uses curly terms for parsed JSON objects, dashes for parsed JSON pairs, and atoms for parsed JSON strings.

Availability:

```
logtalk_load(json(loader))
```

Author: Paulo Moura and Jacinto Dávila
Version: 1:1:0
Date: 2022-11-14

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *json(curly,dash,atom)*

Remarks:

(none)

Inherited public predicates:

generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.165.2 json(StringRepresentation)

- StringRepresentation - Text representation to be used when decoding JSON strings. Possible values are atom (default), chars, and codes.

JSON parser and generator. Uses curly terms for parsed JSON objects and dashes for parsed JSON pairs.

Availability:
logtalk_load(json(loader))

Author: Paulo Moura
Version: 1:0:0
Date: 2022-11-14

Compilation flags:
static, context_switching_calls

Extends:
public *json(curly,dash,StringRepresentation)*

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.165.3 `json(ObjectRepresentation,PairRepresentation,StringRepresentation)`

- `ObjectRepresentation` - Object representation to be used when decoding JSON objects. Possible values are `curly` (default) and `list`.
- `PairRepresentation` - Pair representation to be used when decoding JSON objects. Possible values are `dash` (default), `equal`, and `colon`.
- `StringRepresentation` - Text representation to be used when decoding JSON strings. Possible values are `atom` (default), `chars`, and `codes`.

JSON parser and generator.

Availability:
`logtalk_load(json(loader))`

Author: Paulo Moura and Jacinto Dávila
Version: 1:0:0
Date: 2026-07-05

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *json_protocol*
Uses:
reader

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.165.4 `json_protocol`

JSON parser and generator protocol.

Availability:

```
logtalk_load(json(loader))
```

Author: Paulo Moura and Jacinto Dávila
Version: 1:0:0
Date: 2026-07-05

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses the JSON contents read from the given source (`codes(List)`, `stream(Stream)`, `line(Stream)`, `file(Path)`, `chars(List)`, or `atom(Atom)`) into a term.

Compilation flags:

`static`

Template:

`parse(Source,Term)`

Mode and number of proofs:

`parse(++compound,--term)` - `one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid JSON source term:

`domain_error(json_source,Source)`

`Source` is a valid JSON source term but does not contain parsable JSON:

`domain_error(json,Source)`

---

generate/2

Generates the content using the representation specified in the first argument (`codes(List)`, `stream(Stream)`, `file(Path)`, `chars(List)`, or `atom(Atom)`) for the term in the second argument. Fails if this term cannot be processed.

Compilation flags:

`static`

Template:

`generate(Sink,Term)`

Mode and number of proofs:

`generate(+compound,++term)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid JSON sink term:

`domain_error(json_sink,Sink)`

A JSON version of `Term` cannot be generated by the implementation:

`domain_error(json_term,Term)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.166 json_graph

object

1.166.1 `json_graph`

JSON Graph Format v2 parser and generator.

Availability:
`logtalk_load(json_graph(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *json_graph_protocol*
Uses:
json
list
user

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.166.2 `json_graph_data_protocol`

Canonical JSON graph data protocol for object-backed graph sources and sinks using atom identifiers.

Availability:
`logtalk_load(json_graph(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-03

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *graph/2*
  - *node/3*
  - *edge/5*
  - *hyperedge/4*
  - *hyperedge/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

graph/2

Enumerates or stores graph properties for an atom graph identifier.

Compilation flags:
`dynamic`

Template:
`graph(GraphId,Properties)`
Mode and number of proofs:
`graph(?atom,?list)` - `zero_or_more`

---

node/3

Enumerates or stores node properties for atom graph and node identifiers.

Compilation flags:
`dynamic`

Template:
`node(GraphId,NodeId,Properties)`
Mode and number of proofs:
`node(?atom,?atom,?list)` - `zero_or_more`

---

edge/5

Enumerates or stores edge properties for atom graph, edge, source node, and target node identifiers.

Compilation flags:
`dynamic`

Template:
`edge(GraphId,EdgeId,Source,Target,Properties)`
Mode and number of proofs:
`edge(?atom,?atom,?atom,?atom,?list)` - `zero_or_more`

---

hyperedge/4

Enumerates or stores undirected hyperedge properties for atom graph and hyperedge identifiers and a list of atom node identifiers.

Compilation flags:
`dynamic`

Template:
`hyperedge(GraphId,HyperedgeId,Nodes,Properties)`
Mode and number of proofs:
`hyperedge(?atom,?atom,?list(atom),?list)` - `zero_or_more`

---

hyperedge/5

Enumerates or stores directed hyperedge properties for atom graph and hyperedge identifiers and lists of atom source and target node identifiers.

Compilation flags:
`dynamic`

Template:
`hyperedge(GraphId,HyperedgeId,Sources,Targets,Properties)`
Mode and number of proofs:
`hyperedge(?atom,?atom,?list(atom),?list(atom),?list)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.166.3 json_graph_protocol

JSON Graph Format v2 parser and generator protocol.

Availability:
`logtalk_load(json_graph(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-07-03

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses the graph contents read from the given source (`codes(List)`, `stream(Stream)`, `file(Path)`, `chars(List)`, `atom(Atom)`, or `object(Object)`) into a list of canonical graph terms. Fails if the contents cannot be parsed.

Compilation flags:
`static`

Template:
`parse(Source,Terms)`
Mode and number of proofs:
`parse(++compound,--list)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid JSON graph source term:
`domain_error(json_graph_source,Source)`
`Source` is `object(Object)` but `Object` does not conform to the graph data protocol:
`domain_error(json_graph_source,Source)`

generate/2

Generates the graph contents using the representation specified in the first argument (`codes(List)`, `stream(Stream)`, `file(Path)`, `chars(List)`, `atom(Atom)`, or `object(Object)`) for the list of canonical graph terms in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,Terms)`
Mode and number of proofs:
`generate(+compound,++list)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid JSON graph sink term:
`domain_error(json_graph_sink,Sink)`
`Terms` is neither a variable nor a valid list of canonical graph terms:
`domain_error(json_graph_terms,Terms)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.167 json_ld

object

1.167.1 json_ld

JSON-LD 1.1 parser, generator, and processor. Uses curly terms for parsed JSON objects, dashes for parsed JSON pairs, and atoms for parsed JSON strings.

Availability:
`logtalk_load(json_ld(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-05

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *json_ld(curly,dash,atom)*

Remarks:
(none)

Inherited public predicates:
compact/3 *expand/2* *flatten/2* *frame/3* *generate/2* *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.167.2 `json_ld(StringRepresentation)`

- `StringRepresentation` - Text representation to be used when decoding JSON strings. Possible values are `atom` (default), `chars`, and `codes`.

JSON-LD 1.1 parser, generator, and processor. Uses curly terms for parsed JSON objects and dashes for parsed JSON pairs.

Availability:
`logtalk_load(json_ld(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-05

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *json_ld(curly,dash,StringRepresentation)*

Remarks:
(none)

Inherited public predicates:
compact/3 *expand/2* *flatten/2* *frame/3* *generate/2* *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.167.3 `json_ld(ObjectRepresentation,PairRepresentation,StringRepresentation)`

- `ObjectRepresentation` - Object representation to be used when decoding JSON objects. Possible values are `curly` (default) and `list`.
- `PairRepresentation` - Pair representation to be used when decoding JSON objects. Possible values are `dash` (default), `equal`, and `colon`.
- `StringRepresentation` - Text representation to be used when decoding JSON strings. Possible values are `atom` (default), `chars`, and `codes`.

JSON-LD 1.1 parser, generator, and processor. Builds on top of the `json` library for JSON parsing and generation.

Availability:
`logtalk_load(json_ld(loader))`

Author: Paulo Moura
Version: 1:5:0
Date: 2026-11-04

Compilation flags:

`static, context_switching_calls`

Implements:
`public` *json_ld_protocol*
Uses:
json(ObjectRepresentation,PairRepresentation,StringRepresentation)
list
user

Remarks:
(none)

Inherited public predicates:
compact/3 *expand/2* *flatten/2* *frame/3* *generate/2* *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.167.4 `json_ld_protocol`

JSON-LD 1.1 parser, generator, and processor protocol.

Availability:
`logtalk_load(json_ld(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-08

Compilation flags:

`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
  - *expand/2*
  - *compact/3*
  - *flatten/2*
  - *frame/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a JSON-LD document from the given source (`file(Path)`, `stream(Stream)`, `codes(List)`, `chars(List)`, or `atom(Atom)`) into a term.

Compilation flags:
`static`

Template:
`parse(Source,Term)`

Mode and number of proofs:
`parse(++compound,--term)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(json_ld_source,Source)`

generate/2

Generates the content using the representation specified in the first argument (`file(Path)`, `stream(Stream)`, `codes(List)`, `chars(List)`, or `atom(Atom)`) for the term in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,Term)`
Mode and number of proofs:
`generate(+compound,++term)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` cannot be generated:
`domain_error(json_ld_sink,Sink)`

---

expand/2

Expands a parsed JSON-LD document. Expansion removes the context and represents all properties and types as full IRIs. The result is a list of node objects in expanded document form.

Compilation flags:
`static`

Template:
`expand(Document,Expanded)`
Mode and number of proofs:
`expand(+term,--list)` - `one_or_error`

Exceptions:
`Document` is a variable:
`instantiation_error`

---

compact/3

Compacts an expanded JSON-LD document using the given context. Compaction applies the context to shorten IRIs to terms or compact IRIs.

Compilation flags:
`static`

Template:

```
compact(Document,Context,Compacted)
```

Mode and number of proofs:

```
compact(+term,+term,--term) - one_or_error
```

Exceptions:

`Document` is a variable:

```
instantiation_error
```

`Context` is a variable:

```
instantiation_error
```

---

flatten/2

Flattens an expanded JSON-LD document. Flattening collects all node objects into a flat `@graph` array, with nested nodes replaced by references. Blank node identifiers are generated for nodes without `@id`.

Compilation flags:

```
static
```

Template:

```
flatten(Document,Flattened)
```

Mode and number of proofs:

```
flatten(+term,--term) - one_or_error
```

Exceptions:

`Document` is a variable:

```
instantiation_error
```

---

frame/3

Frames a JSON-LD document using the given frame. The result is an expanded framed document unless the frame provides a context, in which case the result is compacted with that context.

Compilation flags:

```
static
```

Template:

```
frame(Document,Frame,Framed)
```

Mode and number of proofs:

```
frame(+term,+term,--term) - one_or_error
```

Exceptions:

`Document` is a variable:

```
instantiation_error
```

`Frame` is a variable:

```
instantiation_error
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.168 json_lines

object

1.168.1 `json_lines`

JSON Lines parser and generator. Uses curly terms for parsed JSON objects, dashes for parsed JSON pairs, and atoms for parsed JSON strings.

Availability:
`logtalk_load(json_lines(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2025-05-27

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *json_lines(curly,dash,atom)*

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.168.2 `json_lines(StringRepresentation)`

- `StringRepresentation` - Text representation to be used when decoding JSON strings. Possible values are `atom` (default), `chars`, and `codes`.

JSON Lines parser and generator. Uses curly terms for parsed JSON objects and dashes for parsed JSON pairs.

Availability:

```
logtalk_load(json_lines(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2025-05-27

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *json_lines(curly,dash,StringRepresentation)*

Remarks:

(none)

Inherited public predicates:

generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.168.3 `json_lines(ObjectRepresentation,PairRepresentation,StringRepresentation)`

- `ObjectRepresentation` - Object representation to be used when decoding JSON objects. Possible values are `curly` (default) and `list`.
- `PairRepresentation` - Pair representation to be used when decoding JSON objects. Possible values are `dash` (default), `equal`, and `colon`.
- `StringRepresentation` - Text representation to be used when decoding JSON strings. Possible values are `atom` (default), `chars`, and `codes`.

JSON Lines parser and generator.

Availability:
`logtalk_load(json_lines(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-07-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *json_lines_protocol*
Uses:
reader

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.168.4 `json_lines_protocol`

JSON Lines parser and generator protocol.

Availability:
`logtalk_load(json_lines(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses the JSON Lines contents read from the given source (`file(Path)`, `stream(Stream)`, `line(Stream)`, `codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) into a list of ground terms.

Compilation flags:
`static`

Template:
`parse(Source,Terms)`
Mode and number of proofs:
`parse(++compound,--list(ground))` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(json_lines_source,Source)`
`Source` contents are not valid JSON Lines:
`domain_error(json_lines,Source)`

---

generate/2

Generates the content using the representation specified in the first argument (`file(Path)`, `stream(Stream)`, `codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) for the list of ground terms in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,Terms)`
Mode and number of proofs:
`generate(+compound,++list(ground))` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Terms` is a variable:
`instantiation_error`
`Terms` is neither a variable nor a list:

`type_error(list,Terms)`
`Term` is a non-ground element of the list `Terms`:
`instantiation_error`
`Term` is an element of the list `Terms` but not a valid JSON term:
`domain_error(json_term,Term)`
`Sink` cannot be generated:
`domain_error(json_lines_sink,Sink)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.169 json_patch

object

1.169.1 `json_patch`

JSON Patch (RFC 6902) application predicates for native Logtalk JSON terms.

Availability:
`logtalk_load(json_patch(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *json_patch_protocol*
Uses:
json_pointer
list

Remarks:
(none)

Inherited public predicates:

apply/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.169.2 `json_patch_protocol`

JSON Patch (RFC 6902) application protocol.

Availability:
`logtalk_load(json_patch(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-03

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *apply/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

apply/3

Applies a JSON Patch document represented as a list of operation objects to a JSON term and returns the patched JSON term.

Compilation flags:
`static`

Template:
`apply(Patch,OldJSON,NewJSON)`

Mode and number of proofs:
`apply(++list,++term,--term)` - `zero_or_one_or_error`

Exceptions:
`Patch` is a variable:
`instantiation_error`
`OldJSON` is a variable:
`instantiation_error`
`Patch` is neither a variable nor a list:
`type_error(list,Patch)`
An operation in `Patch` is malformed:
`domain_error(json_patch_operation,Operation)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.170 json_path

object

1.170.1 `json_path`

JSONPath implementation using atoms for text representation.

Availability:
`logtalk_load(json_path(loader))`

Author: Paulo Moura
Version: 0:1:0
Date: 2026-07-03

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *json_path(atom)*

Remarks:
(none)

Inherited public predicates:
evaluate/3 *generate/2* *nodes/3* *parse/2* *paths/3* *query/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.170.2 `json_path(StringRepresentation)`

- `StringRepresentation` - Text representation to be used for query member names and normalized paths. Possible values are `atom` (default), `chars`, and `codes`.

Initial JSONPath implementation supporting child and descendant navigation, names, wildcards, indexes, and slices.

Availability:
`logtalk_load(json_path(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2026-11-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *json_path_protocol*
Uses:
list

Remarks:
(none)

Inherited public predicates:
evaluate/3 *generate/2* *nodes/3* *parse/2* *paths/3* *query/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.170.3 `json_path_protocol`

JSONPath parser, generator, and evaluator protocol.

Availability:
`logtalk_load(json_path(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-03

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
  - *evaluate/3*
  - *paths/3*
  - *nodes/3*
  - *query/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a JSONPath query from the given source (`codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) into a parsed query term. Fails if the query cannot be parsed.

Compilation flags:
`static`

Template:
`parse(Source,Query)`
Mode and number of proofs:
`parse(++compound,--compound)` - `zero_or_one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(json_path_source,Source)`

---

generate/2

Generates a JSONPath query using the representation specified in the first argument (`codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) for the parsed query term in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,Query)`
Mode and number of proofs:
`generate(+compound,++compound)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Query` is neither a variable nor a valid parsed query term:
`domain_error(json_path_query,Query)`
`Sink` cannot be generated:
`domain_error(json_path_sink,Sink)`

---

evaluate/3

Evaluates a parsed JSONPath query against a JSON term and returns the selected values.

Compilation flags:
`static`

Template:
`evaluate(Query,JSON,Values)`
Mode and number of proofs:
`evaluate(++compound,++term,--list)` - `zero_or_one_or_error`

Exceptions:
`Query` is a variable:
`instantiation_error`
`JSON` is a variable:
`instantiation_error`
`Query` is neither a variable nor a valid parsed query term:
`domain_error(json_path_query,Query)`

---

paths/3

Evaluates a parsed JSONPath query against a JSON term and returns the selected normalized paths.

Compilation flags:
`static`

Template:
`paths(Query,JSON,Paths)`
Mode and number of proofs:
`paths(++compound,++term,--list)` - `zero_or_one_or_error`

Exceptions:
`Query` is a variable:
`instantiation_error`
`JSON` is a variable:
`instantiation_error`
`Query` is neither a variable nor a valid parsed query term:
`domain_error(json_path_query,Query)`

---

nodes/3

Evaluates a parsed JSONPath query against a JSON term and returns the selected nodes as `node(Path, Value)` terms.

Compilation flags:

`static`

Template:

`nodes(Query,JSON,Nodes)`

Mode and number of proofs:

`nodes(++compound,++term,--list)` - `zero_or_one_or_error`

Exceptions:

`Query` is a variable:

`instantiation_error`

`JSON` is a variable:

`instantiation_error`

`Query` is neither a variable nor a valid parsed query term:

`domain_error(json_path_query,Query)`

---

query/3

Parses a JSONPath query source and evaluates it against a JSON term, returning the selected values.

Compilation flags:

`static`

Template:

`query(Source,JSON,Values)`

Mode and number of proofs:

`query(++compound,++term,--list)` - `zero_or_one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`JSON` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid source:

`domain_error(json_path_source,Source)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.171 json_pointer

object

1.171.1 `json_pointer`

JSON Pointer (RFC 6901) and Relative JSON Pointer parser, generator, and evaluator using atoms for text representation.

Availability:
`logtalk_load(json_pointer(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-04

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *json_pointer(atom)*

Remarks:
(none)

Inherited public predicates:
evaluate/3 *evaluate_relative/4* *generate/2* *generate_fragment/2* *generate_relative/2* *parse/2* *parse_fragment/2* *parse_relative/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.171.2 `json_pointer(StringRepresentation)`

- `StringRepresentation` - Text representation to be used for reference tokens. Possible values are `atom` (default), `chars`, and `codes`.

JSON Pointer (RFC 6901) and Relative JSON Pointer parser, generator, and evaluator.

Availability:
`logtalk_load(json_pointer(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-09-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *json_pointer_protocol*
Uses:
list
utf_8

Remarks:
(none)

Inherited public predicates:
evaluate/3 *evaluate_relative/4* *generate/2* *generate_fragment/2* *generate_relative/2* *parse/2* *parse_fragment/2* *parse_relative/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.171.3 `json_pointer_protocol`

JSON Pointer (RFC 6901) and Relative JSON Pointer parser, generator, and evaluator protocol.

Availability:
`logtalk_load(json_pointer(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-04

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
  - *parse_fragment/2*
  - *generate_fragment/2*
  - *parse_relative/2*
  - *generate_relative/2*
  - *evaluate/3*
  - *evaluate_relative/4*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a JSON Pointer from the given source (`codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) into a list of reference tokens. Fails if the pointer cannot be parsed.

Compilation flags:
`static`

Template:
`parse(Source,Pointer)`
Mode and number of proofs:
`parse(++compound,--list(ground))` - `zero_or_one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(json_pointer_source,Source)`

---

generate/2

Generates a JSON Pointer using the representation specified in the first argument (`codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) for the list of reference tokens in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,Pointer)`
Mode and number of proofs:
`generate(+compound,++list(ground))` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Pointer` is neither a variable nor a list:
`type_error(list,Pointer)`
`Pointer` contains an invalid reference token:
`domain_error(json_pointer_token,Token)`
`Sink` cannot be generated:

```
domain_error(json_pointer_sink,Sink)
```

---

parse_fragment/2

Parses a URI fragment representation of a JSON Pointer from the given source (`codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) into a list of reference tokens. Fails if the fragment cannot be parsed.

Compilation flags:

`static`

Template:

`parse_fragment(Source,Pointer)`

Mode and number of proofs:

`parse_fragment(++compound,--list(ground))` - `zero_or_one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid source:

`domain_error(json_pointer_fragment_source,Source)`

---

generate_fragment/2

Generates the URI fragment representation of a JSON Pointer using the representation specified in the first argument (`codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) for the list of reference tokens in the second argument.

Compilation flags:

`static`

Template:

`generate_fragment(Sink,Pointer)`

Mode and number of proofs:

`generate_fragment(+compound,++list(ground))` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Pointer` is neither a variable nor a list:

`type_error(list,Pointer)`

`Pointer` contains an invalid reference token:

`domain_error(json_pointer_token,Token)`

`Sink` cannot be generated:

`domain_error(json_pointer_fragment_sink,Sink)`

---

parse_relative/2

Parses a Relative JSON Pointer from the given source (`codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) into a canonical term `relative(Up, Shift, Suffix)` where `Up` is the number of ancestor steps, `Shift` is the optional array index adjustment, and `Suffix` is either a list of reference tokens or the atom `'#'`.

Compilation flags:

`static`

Template:

`parse_relative(Source,RelativePointer)`

Mode and number of proofs:

`parse_relative(++compound,--compound)` - `zero_or_one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid source:

`domain_error(json_relative_pointer_source,Source)`

---

generate_relative/2

Generates a Relative JSON Pointer using the representation specified in the first argument (`codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) from a canonical term `relative(Up, Shift, Suffix)` where `Suffix` is either a list of reference tokens or the atom `'#'`.

Compilation flags:

`static`

Template:

`generate_relative(Sink,RelativePointer)`

Mode and number of proofs:

`generate_relative(+compound,++compound)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`RelativePointer` is not a valid Relative JSON Pointer term:

`domain_error(json_relative_pointer,RelativePointer)`

`RelativePointer` contains an invalid reference token:

`domain_error(json_pointer_token,Token)`

`Sink` cannot be generated:

`domain_error(json_relative_pointer_sink,Sink)`

---

evaluate/3

Evaluates a parsed JSON Pointer against a JSON term. Fails if the pointer does not identify a value in the given JSON term.

Compilation flags:
`static`

Template:
`evaluate(Pointer,JSON,Value)`
Mode and number of proofs:
`evaluate(++list(ground),++term,?term)` - `zero_or_one_or_error`

Exceptions:
`Pointer` is a variable:
`instantiation_error`
`JSON` is a variable:
`instantiation_error`
`Pointer` is neither a variable nor a list:
`type_error(list,Pointer)`
`Pointer` contains an invalid reference token:
`domain_error(json_pointer_token,Token)`

---

evaluate_relative/4

Evaluates a parsed Relative JSON Pointer term against a JSON term, using the second argument as the absolute JSON Pointer to the current context value. Fails if either the context or the relative pointer does not identify a value in the given JSON term.

Compilation flags:
`static`

Template:
`evaluate_relative(RelativePointer,Context,JSON,Value)`
Mode and number of proofs:
`evaluate_relative(++compound,++list(ground),++term,?term)` - `zero_or_one_or_error`

Exceptions:
`RelativePointer` is a variable:
`instantiation_error`
`Context` is a variable:
`instantiation_error`
`JSON` is a variable:
`instantiation_error`
`RelativePointer` is not a valid Relative JSON Pointer term:
`domain_error(json_relative_pointer,RelativePointer)`
`Context` is neither a variable nor a list:
`type_error(list,Context)`
`Context` or `RelativePointer` contains an invalid reference token:

```
domain_error(json_pointer_token,Token)
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.172 json_rpc

object

1.172.1 `json_rpc`

JSON-RPC 2.0 protocol encoding and decoding. Provides predicates for constructing and parsing JSON-RPC 2.0 request, notification, response, and error objects. Uses the `json` library for JSON parsing and generation.

Availability:
`logtalk_load(json_rpc(loader))`

Author: Paulo Moura
Version: 1:0:4
Date: 2026-07-13

Compilation flags:
`static, context_switching_calls`

Uses:
json
list
reader

Remarks:

- Specification: Implements the JSON-RPC 2.0 specification (https://www.jsonrpc.org/specification).
- JSON representation: Uses the `json` library default representation: curly terms for objects, dashes for pairs, and atoms for strings.
- Request: A JSON-RPC 2.0 request is represented as `{jsonrpc-'2.0', method-Method, params-Params, id-Id}`.
- Notification: A JSON-RPC 2.0 notification is represented as `{jsonrpc-'2.0', method-Method, params-Params}`.
- Successful response: A JSON-RPC 2.0 successful response is represented as `{jsonrpc-'2.0', result-Result, id-Id}`.

- Error response: A JSON-RPC 2.0 error response is represented as `{jsonrpc-'2.0', error-{code-Code, message-Message}, id-Id}` or `{jsonrpc-'2.0', error-{code-Code, message-Message, data-Data}, id-Id}`.
- Batch request: A JSON-RPC 2.0 batch request is represented as a list of request and/or notification terms.
- Error codes: Standard error codes: -32700 (parse error), -32600 (invalid request), -32601 (method not found), -32602 (invalid params), -32603 (internal error). Server errors: -32000 to -32099.

Inherited public predicates:

(none)

- *Public predicates*
  - *request/4*
  - *request/3*
  - *notification/3*
  - *notification/2*
  - *response/3*
  - *error_response/4*
  - *error_response/5*
  - *parse_error/1*
  - *invalid_request/1*
  - *method_not_found/2*
  - *invalid_params/2*
  - *internal_error/2*
  - *encode/2*
  - *decode/2*
  - *is_request/1*
  - *is_notification/1*
  - *is_response/1*
  - *is_error_response/1*
  - *is_batch/1*
  - *id/2*
  - *method/2*
  - *params/2*
  - *result/2*
  - *error/2*
  - *error_code/2*
  - *error_message/2*
  - *error_data/2*
  - *write_message/2*
  - *read_message/2*
  - *write_framed_message/2*
  - *read_framed_message/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

request/4

Constructs a JSON-RPC 2.0 request term.

Compilation flags:
```
static
```

Template:
```
request(Method,Params,Id,Request)
```
Mode and number of proofs:
```
request(+atom,+list,+nonvar,--compound) - one
```

---

request/3

Constructs a JSON-RPC 2.0 request term with no parameters.

Compilation flags:
```
static
```

Template:
```
request(Method,Id,Request)
```
Mode and number of proofs:
```
request(+atom,+nonvar,--compound) - one
```

---

notification/3

Constructs a JSON-RPC 2.0 notification term (a request without an id).

Compilation flags:
```
static
```

Template:
```
notification(Method,Params,Notification)
```
Mode and number of proofs:
```
notification(+atom,+list,--compound) - one
```

---

notification/2

Constructs a JSON-RPC 2.0 notification term with no parameters.

Compilation flags:
```
static
```

Template:
```
notification(Method,Notification)
```
Mode and number of proofs:
```
notification(+atom,--compound) - one
```

---

response/3

Constructs a JSON-RPC 2.0 successful response term.

Compilation flags:
```
static
```

Template:
```
response(Result,Id,Response)
```
Mode and number of proofs:
```
response(+nonvar,+nonvar,--compound) - one
```

---

error_response/4

Constructs a JSON-RPC 2.0 error response term with a null id (used when the request id cannot be determined).

Compilation flags:
```
static
```

Template:
```
error_response(Code,Message,Id,ErrorResponse)
```
Mode and number of proofs:
```
error_response(+integer,+atom,+nonvar,--compound) - one
```

---

error_response/5

Constructs a JSON-RPC 2.0 error response term with additional error data.

Compilation flags:

```
static
```

Template:

```
error_response(Code,Message,Data,Id,ErrorResponse)
```

Mode and number of proofs:

```
error_response(+integer,+atom,+nonvar,+nonvar,--compound) - one
```

---

parse_error/1

Constructs a JSON-RPC 2.0 parse error response (-32700) with a null id.

Compilation flags:

```
static
```

Template:

```
parse_error(ErrorResponse)
```

Mode and number of proofs:

```
parse_error(--compound) - one
```

---

invalid_request/1

Constructs a JSON-RPC 2.0 invalid request error response (-32600) with a null id.

Compilation flags:

```
static
```

Template:

```
invalid_request(ErrorResponse)
```

Mode and number of proofs:

```
invalid_request(--compound) - one
```

---

method_not_found/2

Constructs a JSON-RPC 2.0 method not found error response (-32601).

Compilation flags:
```
static
```

Template:
```
method_not_found(Id,ErrorResponse)
```
Mode and number of proofs:
```
method_not_found(+nonvar,--compound) - one
```

---

invalid_params/2

Constructs a JSON-RPC 2.0 invalid params error response (-32602).

Compilation flags:
```
static
```

Template:
```
invalid_params(Id,ErrorResponse)
```
Mode and number of proofs:
```
invalid_params(+nonvar,--compound) - one
```

---

internal_error/2

Constructs a JSON-RPC 2.0 internal error response (-32603).

Compilation flags:
```
static
```

Template:
```
internal_error(Id,ErrorResponse)
```
Mode and number of proofs:
```
internal_error(+nonvar,--compound) - one
```

---

encode/2

Encodes a JSON-RPC 2.0 term (request, notification, response, error, or batch) into a JSON atom.

Compilation flags:
`static`

Template:
`encode(Term,JSON)`
Mode and number of proofs:
`encode(+compound,--atom)` - `one`

---

decode/2

Decodes a JSON atom into a JSON-RPC 2.0 term (request, notification, response, error, or batch).

Compilation flags:
`static`

Template:
`decode(JSON,Term)`
Mode and number of proofs:
`decode(+atom,--compound)` - `one_or_error`

Exceptions:
`JSON` is a variable:
`instantiation_error`
`JSON` is neither a variable nor a valid JSON atom term or does not contain parsable JSON by the implementation:
`domain_error(json_source,JSON)`

---

is_request/1

True if the term is a valid JSON-RPC 2.0 request (has jsonrpc, method, and id fields).

Compilation flags:
`static`

Template:
`is_request(Term)`
Mode and number of proofs:
`is_request(+compound)` - `zero_or_one`

---

is_notification/1

True if the term is a valid JSON-RPC 2.0 notification (has jsonrpc and method fields but no id field).

Compilation flags:
`static`

Template:
`is_notification(Term)`
Mode and number of proofs:
`is_notification(+compound)` - `zero_or_one`

---

is_response/1

True if the term is a valid JSON-RPC 2.0 successful response (has jsonrpc, result, and id fields).

Compilation flags:
`static`

Template:
`is_response(Term)`
Mode and number of proofs:
`is_response(+compound)` - `zero_or_one`

---

is_error_response/1

True if the term is a valid JSON-RPC 2.0 error response (has jsonrpc, error, and id fields).

Compilation flags:
`static`

Template:
`is_error_response(Term)`
Mode and number of proofs:
`is_error_response(+compound)` - `zero_or_one`

---

is_batch/1

True if the term is a valid JSON-RPC 2.0 batch (a non-empty list).

Compilation flags:
```
static
```

Template:
```
is_batch(Term)
```
Mode and number of proofs:
```
is_batch(+compound) - zero_or_one
```

---

id/2

Extracts the id field from a JSON-RPC 2.0 message.

Compilation flags:
```
static
```

Template:
```
id(Message,Id)
```
Mode and number of proofs:
```
id(+compound,--nonvar) - zero_or_one
```

---

method/2

Extracts the method field from a JSON-RPC 2.0 request or notification.

Compilation flags:
```
static
```

Template:
```
method(Message,Method)
```
Mode and number of proofs:
```
method(+compound,--atom) - zero_or_one
```

---

params/2

Extracts the params field from a JSON-RPC 2.0 request or notification.

Compilation flags:
`static`

Template:
`params(Message,Params)`
Mode and number of proofs:
`params(+compound,--nonvar)` - `zero_or_one`

---

result/2

Extracts the result field from a JSON-RPC 2.0 response.

Compilation flags:
`static`

Template:
`result(Message,Result)`
Mode and number of proofs:
`result(+compound,--nonvar)` - `zero_or_one`

---

error/2

Extracts the error field from a JSON-RPC 2.0 error response.

Compilation flags:
`static`

Template:
`error(Message,Error)`
Mode and number of proofs:
`error(+compound,--compound)` - `zero_or_one`

---

error_code/2

Extracts the error code from a JSON-RPC 2.0 error response.

Compilation flags:

```
static
```

Template:

```
error_code(Message,Code)
```

Mode and number of proofs:

```
error_code(+compound,--integer) - zero_or_one
```

---

error_message/2

Extracts the error message from a JSON-RPC 2.0 error response.

Compilation flags:

```
static
```

Template:

```
error_message(Message,ErrorMessage)
```

Mode and number of proofs:

```
error_message(+compound,--atom) - zero_or_one
```

---

error_data/2

Extracts the error data from a JSON-RPC 2.0 error response. Fails if no data field is present.

Compilation flags:

```
static
```

Template:

```
error_data(Message,Data)
```

Mode and number of proofs:

```
error_data(+compound,--nonvar) - zero_or_one
```

---

write_message/2

Writes a JSON-RPC 2.0 message to an output text stream as a single line of JSON followed by a newline. Flushes the output stream after writing.

Compilation flags:
`static`

Template:
`write_message(Output,Message)`
Mode and number of proofs:
`write_message(+stream,+compound)` - `one`

---

read_message/2

Reads a JSON-RPC 2.0 message from an input text stream. Reads a line of JSON text and parses it. Fails at end of stream.

Compilation flags:
`static`

Template:
`read_message(Input,Message)`
Mode and number of proofs:
`read_message(+stream,--compound)` - `zero_or_one`

---

write_framed_message/2

Writes a JSON-RPC 2.0 message to an output stream using Content-Length framing (as used by LSP style protocols). The message is preceded by a `Content-Length:  N\r\n\r\n` header where `N` is the byte length of the JSON body. Flushes the output stream after writing.

Compilation flags:
`static`

Template:
`write_framed_message(Output,Message)`
Mode and number of proofs:
`write_framed_message(+stream,+compound)` - `one`

---

read_framed_message/2

Reads a JSON-RPC 2.0 message from an input stream using Content-Length framing (as used by LSP style protocols). Reads a `Content-Length: N\r\n\r\n` header followed by exactly `N` bytes of JSON body. Fails at end of stream or if the header is missing or malformed.

Compilation flags:
`static`

Template:
`read_framed_message(Input,Message)`
Mode and number of proofs:
`read_framed_message(+stream,--compound)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.173 json_schema

object

1.173.1 json_schema

JSON Schema parser and validator. Uses curly terms for JSON objects, dashes for JSON pairs, and atoms for JSON strings.

Availability:
`logtalk_load(json_schema(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-01-29

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *json_schema(curly,dash,atom)*

Remarks:

(none)

Inherited public predicates:

parse/2 *validate/2* *validate/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.173.2 `json_schema(StringRepresentation)`

- `StringRepresentation` - String representation used for JSON strings. Possible values are `atom` (default), `chars`, and `codes`.

JSON Schema parser and validator. Uses curly terms for JSON objects and dashes for JSON pairs.

Availability:

`logtalk_load(json_schema(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-01-29

Compilation flags:

`static, context_switching_calls`

Extends:

`public` *json_schema(curly,dash,StringRepresentation)*

Remarks:

(none)

Inherited public predicates:

parse/2 *validate/2* *validate/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.173.3 `json_schema(ObjectRepresentation,PairRepresentation,StringRepresentation)`

- `ObjectRepresentation` - Object representation used for JSON objects. Possible values are `curly` (default) and `list`.
- `PairRepresentation` - Pair representation used for JSON object pairs. Possible values are `dash` (default), `equal`, and `colon`.
- `StringRepresentation` - String representation used for JSON strings. Possible values are `atom` (default), `chars`, and `codes`.

JSON Schema parser and validator supporting JSON Schema draft-07/draft-2019-09/draft-2020-12.

Availability:

`logtalk_load(json_schema(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2026-07-06

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *json_schema_protocol*

Uses:

json(ObjectRepresentation,PairRepresentation,StringRepresentation)

json_pointer(StringRepresentation)
list
os
url(Representation)

Remarks:
(none)

Inherited public predicates:
parse/2 *validate/2* *validate/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.173.4 `json_schema_protocol`

JSON Schema parser and validator protocol.

Availability:
`logtalk_load(json_schema(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-06-14

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *validate/2*
  - *validate/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a JSON schema from the given source (`file(Path)`, `stream(Stream)`, `codes(List)`, `chars(List)`, or `atom(Atom)`) into a schema term.

Compilation flags:
`static`

Template:
`parse(Source,Schema)`

Mode and number of proofs:
`parse(++compound,--term)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid JSON source term or does not contain parsable JSON:
`domain_error(json_source,Source)`
The parsed JSON value is not a valid JSON Schema term:
`domain_error(json_schema,Schema)`

validate/2

Validates a JSON term against a parsed schema. Succeeds if the JSON term is valid according to the schema.

Compilation flags:
`static`

Template:
`validate(Schema,JSON)`
Mode and number of proofs:
`validate(+term,+term)` - `zero_or_one`

---

validate/3

Validates a JSON term against a parsed schema. Returns a list of validation errors (empty list if valid).

Compilation flags:
`static`

Template:
`validate(Schema,JSON,Errors)`
Mode and number of proofs:
`validate(+term,+term,--list)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.174 jwt

object

1.174.1 jwt

Facade predicates for parsing, signing, verifying, and validating JSON Web Tokens.

Availability:
`logtalk_load(jwt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *jwt_helpers*
Uses:
jwt_claims
jwt_compact
jwt_jwa
jwt_jwks
jwt_jws

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *decode/3*
  - *decode/5*
  - *header/2*
  - *claims/2*
  - *claim/3*
  - *peek_algorithm/2*
  - *peek_key_id/2*
  - *verify/4*
  - *verify/5*
  - *verify_signature/5*
  - *sign/5*
  - *sign_payload/5*
  - *validate_claims/3*
  - *validate_claim/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

decode/3

Decodes a compact JWT into header and claims JSON terms without verifying the signature.

Compilation flags:
`static`

Template:
`decode(Token,Header,Claims)`
Mode and number of proofs:
`decode(+atom,-term,-term)` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a valid compact JWT:
`domain_error(jwt_compact_serialization,Token)`
`Token` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

decode/5

Decodes a compact JWT into header, claims, signature bytes, and signing input.

Compilation flags:
`static`

Template:
`decode(Token,Header,Claims,Signature,SigningInput)`
Mode and number of proofs:
`decode(+atom,-term,-term,-list(byte),-atom)` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a valid compact JWT:
`domain_error(jwt_compact_serialization,Token)`
`Token` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

header/2

Decodes only the JWT header JSON term.

Compilation flags:
`static`

Template:
`header(Token,Header)`
Mode and number of proofs:
`header(+atom,-term)` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a valid compact JWT:
`domain_error(jwt_compact_serialization,Token)`
`Token` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

claims/2

Decodes only the JWT claims JSON term.

Compilation flags:
`static`

Template:
`claims(Token,Claims)`
Mode and number of proofs:
`claims(+atom,-term)` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a valid compact JWT:
`domain_error(jwt_compact_serialization,Token)`
`Token` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

claim/3

Looks up a claim value by name in a claims JSON term.

Compilation flags:
`static`

Template:
`claim(Claims,Name,Value)`
Mode and number of proofs:
`claim(+term,+atom,-term)` - `zero_or_one`

---

peek_algorithm/2

Decodes the JWT header and returns its algorithm.

Compilation flags:
`static`

Template:
`peek_algorithm(Token,Algorithm)`
Mode and number of proofs:
`peek_algorithm(+atom,-atom)` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a valid compact JWT:
`domain_error(jwt_compact_serialization,Token)`
`Token` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`
The decoded JWT header does not contain a valid `alg` member:
`domain_error(jwt_header,Header)`

---

peek_key_id/2

Decodes the JWT header and returns its optional key identifier.

Compilation flags:
`static`

Template:
`peek_key_id(Token,KeyId)`

Mode and number of proofs:

`peek_key_id(+atom,-atom)` - `zero_or_one`

`verify/4`

Verifies a compact JWT using a key or JWK Set and returns the claims.

Compilation flags:

`static`

Template:

`verify(Token,KeyOrJWKSet,Claims,Options)`

Mode and number of proofs:

`verify(+atom,+term,-term,+list(compound))` - `zero_or_one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Token` is a variable:

`instantiation_error`

`Token` is neither a variable nor an atom:

`type_error(atom,Token)`

`Token` is an atom but not a valid compact JWT:

`domain_error(jwt_compact_serialization,Token)`

`Token` contains Base64URL data with characters outside the Base64URL alphabet:

`representation_error(base64)`

The decoded JWT header does not contain a valid `alg` member:

`domain_error(jwt_header,Header)`

The JWT algorithm is unsupported or disallowed:

`domain_error(jwt_algorithm,Algorithm)`

`KeyOrJWKSet` is not a JSON Web Key Set with a `keys` list:

`domain_error(jwt_jwks,KeyOrJWKSet)`

No compatible key exists for the JWT header:

`existence_error(jwt_jwk,Header)`

`KeyOrJWKSet` is not a valid symmetric key:

`domain_error(jwt_symmetric_key,KeyOrJWKSet)`

`KeyOrJWKSet` is not a supported public JWK:

`domain_error(jwt_jwk_public_key,KeyOrJWKSet)`

The ES256 signature is not 64 bytes long:

`domain_error(jwt_es256_signature,Signature)`

The OpenSSL executable does not exist:

`existence_error(os_command,Executable)`

A required claim is missing:

`domain_error(jwt_claims,missing(Name))`

A claim has an invalid value:

`domain_error(jwt_claim(Name),Value)`

A time claim value is not a number:

`type_error(time_number,Name-Time)`

---

verify/5

Verifies a compact JWT using a key or JWK Set and returns the header and claims.

Compilation flags:

`static`

Template:

`verify(Token,Header,Claims,KeyOrJWKSet,Options)`

Mode and number of proofs:

`verify(+atom,-term,-term,+term,+list(compound))` - `zero_or_one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Token` is a variable:

`instantiation_error`

`Token` is neither a variable nor an atom:

`type_error(atom,Token)`

`Token` is an atom but not a valid compact JWT:

`domain_error(jwt_compact_serialization,Token)`

`Token` contains Base64URL data with characters outside the Base64URL alphabet:

`representation_error(base64)`

The decoded JWT header does not contain a valid `alg` member:

`domain_error(jwt_header,Header)`

The JWT algorithm is unsupported or disallowed:

`domain_error(jwt_algorithm,Algorithm)`

`KeyOrJWKSet` is not a JSON Web Key Set with a `keys` list:

`domain_error(jwt_jwks,KeyOrJWKSet)`

No compatible key exists for the JWT header:

`existence_error(jwt_jwk,Header)`

`KeyOrJWKSet` is not a valid symmetric key:

`domain_error(jwt_symmetric_key,KeyOrJWKSet)`

`KeyOrJWKSet` is not a supported public JWK:
```
domain_error(jwt_jwk_public_key,KeyOrJWKSet)
```
The ES256 signature is not 64 bytes long:
```
domain_error(jwt_es256_signature,Signature)
```
The OpenSSL executable does not exist:
```
existence_error(os_command,Executable)
```
A required claim is missing:
```
domain_error(jwt_claims,missing(Name))
```
A claim has an invalid value:
```
domain_error(jwt_claim(Name),Value)
```
A time claim value is not a number:
```
type_error(time_number,Name-Time)
```

---

verify_signature/5

Verifies a JWS signature for a signing input, algorithm, key, and options.

Compilation flags:
```
static
```

Template:
```
verify_signature(Algorithm,SigningInput,Signature,Key,Options)
```
Mode and number of proofs:
```
verify_signature(+atom,+atom,+list(byte),+term,+list(compound)) -
zero_or_one_or_error
```

Exceptions:

`Options` is a variable or a partial list:
```
instantiation_error
```
`Options` is neither a variable nor a list:
```
type_error(list,Options)
```
An element `Option` of the list `Options` is neither a variable nor a compound term:
```
type_error(compound,Option)
```
An element `Option` of the list `Options` is a compound term but not a valid option:
```
domain_error(option,Option)
```
`Algorithm` is unsupported or disallowed:
```
domain_error(jwt_algorithm,Algorithm)
```
`Key` is not a valid symmetric key:
```
domain_error(jwt_symmetric_key,Key)
```
`Key` is not a supported public JWK:
```
domain_error(jwt_jwk_public_key,Key)
```
`Signature` is not a 64-byte raw ES256 signature:
```
domain_error(jwt_es256_signature,Signature)
```
The OpenSSL executable does not exist:
```
existence_error(os_command,Executable)
```

---

sign/5

Signs claims as a compact JWT using the given header, key, and options.

Compilation flags:
`static`

Template:
`sign(Header,Claims,Key,Token,Options)`
Mode and number of proofs:
`sign(+term,+term,+term,-atom,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Header` does not contain a valid `alg` member:
`domain_error(jwt_header,Header)`
The JWT algorithm is unsupported or disallowed:
`domain_error(jwt_algorithm,Algorithm)`
`Header` or `Claims` cannot be encoded as JSON:
`domain_error(json_sink,HeaderOrClaims)`
`Key` is not a valid symmetric key:
`domain_error(jwt_symmetric_key,Key)`
`Algorithm` is not a supported signing algorithm:
`domain_error(jwt_signing_algorithm,Algorithm)`

---

sign_payload/5

Signs a payload JSON term as a compact JWS using the given header, key, and options.

Compilation flags:
`static`

Template:
`sign_payload(Header,Payload,Key,Token,Options)`
Mode and number of proofs:
`sign_payload(+term,+term,+term,-atom,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`

`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Header` does not contain a valid `alg` member:
`domain_error(jwt_header,Header)`
The JWT algorithm is unsupported or disallowed:
`domain_error(jwt_algorithm,Algorithm)`
`Header` or `Payload` cannot be encoded as JSON:
`domain_error(json_sink,HeaderOrPayload)`
`Key` is not a valid symmetric key:
`domain_error(jwt_symmetric_key,Key)`
`Algorithm` is not a supported signing algorithm:
`domain_error(jwt_signing_algorithm,Algorithm)`

---

validate_claims/3

Validates JWT claims using registered-claim defaults, a policy list, and options.

Compilation flags:
`static`

Template:
`validate_claims(Claims,Policy,Options)`
Mode and number of proofs:
`validate_claims(+term,+list(compound),+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
A required claim is missing:
`domain_error(jwt_claims,missing(Name))`
A claim has an invalid value:
`domain_error(jwt_claim(Name),Value)`
`Policy` contains an invalid claim policy:
`domain_error(jwt_claim_policy,Policy)`
A time claim value is not a number:
`type_error(time_number,Name-Time)`
`Policy` contains an invalid time-claim kind:

`domain_error(jwt_time_claim_kind,Kind)`

`validate_claim/3`

Validates a single JWT claim policy against a claims JSON term.

Compilation flags:
`static`

Template:
`validate_claim(Claims,ClaimPolicy,Options)`
Mode and number of proofs:
`validate_claim(+term,+compound,+list(compound))` - `one_or_error`

Exceptions:
A required claim is missing:
`domain_error(jwt_claims,missing(Name))`
A claim has an invalid value:
`domain_error(jwt_claim(Name),Value)`
`ClaimPolicy` is not a valid claim policy:
`domain_error(jwt_claim_policy,ClaimPolicy)`
A time claim value is not a number:
`type_error(time_number,Name-Time)`
`ClaimPolicy` contains an invalid time-claim kind:
`domain_error(jwt_time_claim_kind,Kind)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.174.2 jwt_claims

Reusable JWT registered-claim validation helpers.

Availability:
`logtalk_load(jwt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *jwt_helpers*
Uses:
list
os

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *validate_claims/3*
  - *validate_claim/3*
  - *claim/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

validate_claims/3

Validates JWT claims using registered-claim defaults, a policy list, and options.

Compilation flags:
`static`

Template:
`validate_claims(Claims,Policy,Options)`
Mode and number of proofs:
`validate_claims(+term,+list(compound),+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

```
instantiation_error
```

`Options` is neither a variable nor a list:

```
type_error(list,Options)
```

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

A required claim is missing:

```
domain_error(jwt_claims,missing(Name))
```

A claim has an invalid value:

```
domain_error(jwt_claim(Name),Value)
```

`Policy` contains an invalid claim policy:

```
domain_error(jwt_claim_policy,Policy)
```

A time claim value is not a number:

```
type_error(time_number,Name-Time)
```

`Policy` contains an invalid time-claim kind:

```
domain_error(jwt_time_claim_kind,Kind)
```

---

validate_claim/3

Validates a single JWT claim policy against a claims JSON term.

Compilation flags:

```
static
```

Template:

```
validate_claim(Claims,ClaimPolicy,Options)
```

Meta-predicate template:

```
validate_claim(*,*,*)
```

Mode and number of proofs:

```
validate_claim(+term,+compound,+list(compound)) - one_or_error
```

Exceptions:

A required claim is missing:

```
domain_error(jwt_claims,missing(Name))
```

A claim has an invalid value:

```
domain_error(jwt_claim(Name),Value)
```

`ClaimPolicy` is not a valid claim policy:

```
domain_error(jwt_claim_policy,ClaimPolicy)
```

A time claim value is not a number:

```
type_error(time_number,Name-Time)
```

`ClaimPolicy` contains an invalid time-claim kind:

```
domain_error(jwt_time_claim_kind,Kind)
```

`claim/3`

Looks up a claim value by name in a claims JSON term.

Compilation flags:
`static`

Template:
`claim(Claims,Name,Value)`
Mode and number of proofs:
`claim(+term,+atom,-term)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.174.3 `jwt_compact`

JWT compact serialization parsing and generation helpers.

Availability:
`logtalk_load(jwt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *jwt_helpers*
Uses:
atom
json
user

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *decode/3*
  - *decode/5*
  - *header/2*
  - *claims/2*
  - *segments/5*
  - *signing_input/4*
  - *compact/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

decode/3

Decodes a compact JWT into header and claims JSON terms without verifying the signature.

Compilation flags:

`static`

Template:

`decode(Token,Header,Claims)`

Mode and number of proofs:

`decode(+atom,-term,-term)` - `one_or_error`

Exceptions:

`Token` is a variable:

`instantiation_error`

`Token` is neither a variable nor an atom:

`type_error(atom,Token)`

`Token` is an atom but not a valid compact JWT:

`domain_error(jwt_compact_serialization,Token)`

`Token` contains Base64URL data with characters outside the Base64URL alphabet:

`representation_error(base64)`

decode/5

Decodes a compact JWT into header, claims, signature bytes, and signing input.

Compilation flags:
`static`

Template:
`decode(Token,Header,Claims,Signature,SigningInput)`
Mode and number of proofs:
`decode(+atom,-term,-term,-list(byte),-atom)` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a valid compact JWT:
`domain_error(jwt_compact_serialization,Token)`
`Token` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

header/2

Decodes only the JWT header JSON term.

Compilation flags:
`static`

Template:
`header(Token,Header)`
Mode and number of proofs:
`header(+atom,-term)` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a valid compact JWT:
`domain_error(jwt_compact_serialization,Token)`
`Token` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

claims/2

Decodes only the JWT claims JSON term.

Compilation flags:
`static`

Template:
`claims(Token,Claims)`
Mode and number of proofs:
`claims(+atom,-term)` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a valid compact JWT:
`domain_error(jwt_compact_serialization,Token)`
`Token` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

segments/5

Splits a compact JWT into Base64URL segments and signing input.

Compilation flags:
`static`

Template:
`segments(Token,HeaderSegment,PayloadSegment,SignatureSegment,SigningInput)`
Mode and number of proofs:
`segments(+atom,-atom,-atom,-atom,-atom)` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a valid compact JWT:
`domain_error(jwt_compact_serialization,Token)`

---

signing_input/4

Encodes header and payload JSON terms and returns the compact signing input.

Compilation flags:
`static`

Template:
`signing_input(Header,Payload,HeaderSegment,SigningInput)`
Mode and number of proofs:
`signing_input(+term,+term,-atom,-atom)` - `one_or_error`

Exceptions:
`Header` or `Payload` cannot be encoded as JSON:
`domain_error(json_sink,HeaderOrPayload)`
`Header` or `Payload` cannot be encoded as Base64URL bytes:
`domain_error(base64url_sink,codes(Codes))`

---

compact/4

Builds a compact JWT from a signing input and signature bytes.

Compilation flags:
`static`

Template:
`compact(HeaderSegment,SigningInput,Signature,Token)`
Mode and number of proofs:
`compact(+atom,+atom,+list(byte),-atom)` - `one_or_error`

Exceptions:
`Signature` cannot be encoded as Base64URL bytes:
`domain_error(base64url_sink,codes(Codes))`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.174.4 `jwt_der`

DER and PEM helpers for JWK public keys and JOSE signatures.

Availability:
`logtalk_load(jwt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *jwt_helpers*
Uses:
base64
list
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *public_key_pem/2*
  - *es256_signature_der/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

public_key_pem/2

Converts a supported JWK public key into PEM SubjectPublicKeyInfo text.

Compilation flags:
`static`

Template:
`public_key_pem(Key,PEM)`
Mode and number of proofs:
`public_key_pem(+term,-atom)` - `one_or_error`

Exceptions:
`Key` is not a supported RSA or P-256 EC public JWK:
`domain_error(jwt_jwk_public_key,Key)`
`Key` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

es256_signature_der/2

Converts a raw JOSE ES256 signature into DER-encoded ASN.1 form.

Compilation flags:
`static`

Template:
`es256_signature_der(Signature,DER)`
Mode and number of proofs:
`es256_signature_der(+list(byte),-list(byte))` - `one_or_error`

Exceptions:
`Signature` is not a 64-byte raw ES256 signature:
`domain_error(jwt_es256_signature,Signature)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.174.5 `jwt_helpers`

Internal shared helpers for the jwt library.

Availability:
`logtalk_load(jwt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:
`static`

Extends:
`public` *options*
Uses:
base64url_no_padding
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *json_member/3*
  - *json_member_default/4*
  - *json_object_pairs/2*
  - *json_object/1*
  - *pair_key_value/3*
  - *base64url_atom_bytes/2*
  - *atom_bytes/2*
  - *bytes_atom/2*
  - *atom_list/1*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

json_member/3

Enumerates JSON object member values for a given key.

Compilation flags:
`static`

Template:
`json_member(Key,Object,Value)`
Mode and number of proofs:
`json_member(+atom,+term,-term)` - `zero_or_more`

---

json_member_default/4

Returns a JSON object member value or a default when the key is absent.

Compilation flags:
`static`

Template:
`json_member_default(Key,Object,Default,Value)`
Mode and number of proofs:
`json_member_default(+atom,+term,+term,-term)` - `one`

---

json_object_pairs/2

Extracts the key-value pairs from a JSON object term.

Compilation flags:
`static`

Template:
`json_object_pairs(Object,Pairs)`
Mode and number of proofs:
`json_object_pairs(+term,-list(compound))` - `zero_or_one`

---

json_object/1

Validates that a JSON term is an object without duplicate member names.

Compilation flags:
`static`

Template:
`json_object(Object)`
Mode and number of proofs:
`json_object(+term)` - `one_or_error`

Exceptions:
`Object` is not a JSON object or contains duplicate member names:
`domain_error(jwt_json_object,Object)`

---

pair_key_value/3

Extracts the key and value from a JSON pair compound term.

Compilation flags:
`static`

Template:
`pair_key_value(Pair,Key,Value)`
Mode and number of proofs:
`pair_key_value(+compound,-term,-term)` - `one`

---

base64url_atom_bytes/2

Converts between unpadded Base64URL atoms and byte lists.

Compilation flags:
`static`

Template:
`base64url_atom_bytes(Atom,Bytes)`
Mode and number of proofs:
`base64url_atom_bytes(+atom,-list(byte))` - `one_or_error`
`base64url_atom_bytes(-atom,+list(byte))` - `one_or_error`

Exceptions:
`Atom` is a variable or the sink is a variable:
`instantiation_error`
`Atom` is neither a variable nor a valid Base64URL source term:

`domain_error(base64url_source,codes(Codes))`
`Bytes` cannot be encoded in the requested Base64URL sink representation:
`domain_error(base64url_sink,codes(Codes))`
`Atom` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

atom_bytes/2

Converts between an atom and its byte list representation.

Compilation flags:
`static`

Template:
`atom_bytes(Atom,Bytes)`
Mode and number of proofs:
`atom_bytes(+atom,-list(byte))` - `one`
`atom_bytes(-atom,+list(byte))` - `one`

---

bytes_atom/2

Converts a byte list into an atom.

Compilation flags:
`static`

Template:
`bytes_atom(Bytes,Atom)`
Mode and number of proofs:
`bytes_atom(+list(byte),-atom)` - `one`

---

atom_list/1

Validates that a list contains only atoms.

Compilation flags:
`static`

Template:
`atom_list(Atoms)`
Mode and number of proofs:
`atom_list(+list(atom))` - `zero_or_one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.174.6 `jwt_jwa`

JSON Web Algorithm metadata and header helpers.

Availability:
`logtalk_load(jwt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *jwt_helpers*
Uses:
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *header_algorithm/2*
  - *header_key_id/2*
  - *validate_header/1*
  - *allowed_algorithm/2*
  - *supported_algorithm/1*
  - *key_type/2*
  - *hmac_hash/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

header_algorithm/2

Extracts and validates the JWT header algorithm.

Compilation flags:
`static`

Template:
`header_algorithm(Header,Algorithm)`
Mode and number of proofs:
`header_algorithm(+term,-atom)` - `one_or_error`

Exceptions:
`Header` does not contain a valid `alg` member:
`domain_error(jwt_header,Header)`

---

header_key_id/2

Extracts the optional key identifier from a JWT header.

Compilation flags:
`static`

Template:
`header_key_id(Header,KeyId)`
Mode and number of proofs:
`header_key_id(+term,-atom)` - `zero_or_one`

---

validate_header/1

Validates a JWT header object and rejects unsupported critical header parameters.

Compilation flags:
`static`

Template:
`validate_header(Header)`
Mode and number of proofs:
`validate_header(+term)` - `one_or_error`

Exceptions:
`Header` is not a JSON object, contains duplicate member names, or contains unsupported critical header parameters:
`domain_error(jwt_header,Header)`

---

allowed_algorithm/2

Validates that an algorithm is supported and allowed by the options.

Compilation flags:
`static`

Template:
`allowed_algorithm(Algorithm,Options)`
Mode and number of proofs:
`allowed_algorithm(+atom,+list(compound))` - `one_or_error`

Exceptions:
`Algorithm` is unsupported or disallowed by `Options`:
`domain_error(jwt_algorithm,Algorithm)`

---

supported_algorithm/1

Succeeds when the algorithm is supported by this library.

Compilation flags:
`static`

Template:
`supported_algorithm(Algorithm)`
Mode and number of proofs:
`supported_algorithm(+atom)` - `zero_or_one`

---

key_type/2

Maps a supported algorithm to the required JWK key type.

Compilation flags:
`static`

Template:
`key_type(Algorithm,KeyType)`
Mode and number of proofs:
`key_type(+atom,-atom)` - `zero_or_one`

---

hmac_hash/2

Maps a supported HMAC algorithm to its hash object.

Compilation flags:
`static`

Template:
`hmac_hash(Algorithm,Hash)`
Mode and number of proofs:
`hmac_hash(+atom,-object_identifier)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.174.7 jwt_jwk

JSON Web Key normalization and key material helpers.

Availability:
`logtalk_load(jwt(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-07-26

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *jwt_helpers*
Uses:
jwt_der
list

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *public_key_pem/2*
  - *symmetric_key_bytes/2*
  - *compatible_key/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

public_key_pem/2

Converts a supported JWK public key into PEM SubjectPublicKeyInfo text.

Compilation flags:
`static`

Template:
`public_key_pem(Key,PEM)`
Mode and number of proofs:
`public_key_pem(+term,-atom)` - `one_or_error`

Exceptions:
`Key` is not a supported RSA or P-256 EC public JWK:
`domain_error(jwt_jwk_public_key,Key)`
`Key` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

symmetric_key_bytes/2

Extracts symmetric key bytes from an atom, byte wrapper, or octet JWK.

Compilation flags:
`static`

Template:
`symmetric_key_bytes(Key,Bytes)`
Mode and number of proofs:
`symmetric_key_bytes(+term,-list(byte))` - `one_or_error`

Exceptions:
`Key` is not an atom, byte wrapper, or octet JWK with Base64URL key material:
`domain_error(jwt_symmetric_key,Key)`

`Key` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

`compatible_key/2`

Succeeds when a key is compatible with the requested JWT algorithm.

Compilation flags:
`static`

Template:
`compatible_key(Key,Algorithm)`
Mode and number of proofs:
`compatible_key(+term,+atom)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.174.8 `jwt_jwks`

JSON Web Key Set validation and key selection helpers.

Availability:
`logtalk_load(jwt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *jwt_helpers*
Uses:
jwt_jwa

jwt_jwk
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *validate/1*
  - *select_key/3*
  - *select_key/4*
  - *select_keys/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

validate/1

Validates that a term is a JSON Web Key Set.

Compilation flags:
`static`

Template:
`validate(JWKSet)`
Mode and number of proofs:
`validate(+term)` - `one_or_error`

Exceptions:
`JWKSet` is not a JSON Web Key Set with a `keys` list:
`domain_error(jwt_jwks,JWKSet)`

---

select_key/3

Selects a matching verification key from a JWK Set for a JWT header.

Compilation flags:
`static`

Template:
`select_key(JWKSet,Header,Key)`

Mode and number of proofs:

`select_key(+term,+term,-term)` - `one_or_error`

Exceptions:

`JWKSet` is not a JSON Web Key Set with a `keys` list:

`domain_error(jwt_jwks,JWKSet)`

`Header` does not contain a valid `alg` member:

`domain_error(jwt_header,Header)`

The JWT algorithm is unsupported or disallowed:

`domain_error(jwt_algorithm,Algorithm)`

No compatible key exists for `Header`:

`existence_error(jwt_jwk,Header)`

`select_key/4`

Selects a matching verification key from a JWK Set for a JWT header and options.

Compilation flags:

`static`

Template:

`select_key(JWKSet,Header,Key,Options)`

Mode and number of proofs:

`select_key(+term,+term,-term,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`JWKSet` is not a JSON Web Key Set with a `keys` list:

`domain_error(jwt_jwks,JWKSet)`

`Header` does not contain a valid `alg` member:

`domain_error(jwt_header,Header)`

The JWT algorithm is unsupported or disallowed:

`domain_error(jwt_algorithm,Algorithm)`

No compatible key exists for `Header`:

`existence_error(jwt_jwk,Header)`

---

select_keys/4

Selects all matching verification keys from a JWK Set for a JWT header and options.

Compilation flags:
`static`

Template:
`select_keys(JWKSet,Header,Keys,Options)`
Mode and number of proofs:
`select_keys(+term,+term,-list(term),+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`JWKSet` is not a JSON Web Key Set with a `keys` list:
`domain_error(jwt_jwks,JWKSet)`
`Header` does not contain a valid `alg` member:
`domain_error(jwt_header,Header)`
The JWT algorithm is unsupported or disallowed:
`domain_error(jwt_algorithm,Algorithm)`
No compatible key exists for `Header`:
`existence_error(jwt_jwk,Header)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.174.9 `jwt_jws`

JWS signing input construction and signature algorithm dispatch.

Availability:
`logtalk_load(jwt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *jwt_helpers*
Uses:
hmac
jwt_compact
jwt_jwa
jwt_jwk
jwt_openssl

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *verify/5*
  - *sign/5*
  - *sign_payload/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`verify/5`

Verifies a JWS signature for a signing input, algorithm, key, and options.

Compilation flags:
`static`

Template:
`verify(Algorithm,SigningInput,Signature,Key,Options)`

Mode and number of proofs:

`verify(+atom,+atom,+list(byte),+term,+list(compound))` - `zero_or_one_or_error`

Exceptions:

`Algorithm` is unsupported:

`domain_error(jwt_algorithm,Algorithm)`

`Key` is not a valid symmetric key:

`domain_error(jwt_symmetric_key,Key)`

`Key` is not a supported public JWK:

`domain_error(jwt_jwk_public_key,Key)`

`Signature` is not a 64-byte raw ES256 signature:

`domain_error(jwt_es256_signature,Signature)`

The OpenSSL executable does not exist:

`existence_error(os_command,Executable)`

`sign/5`

Signs claims as a compact JWT using the given header, key, and options.

Compilation flags:

`static`

Template:

`sign(Header,Claims,Key,Token,Options)`

Mode and number of proofs:

`sign(+term,+term,+term,-atom,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Header` does not contain a valid `alg` member:

`domain_error(jwt_header,Header)`

The JWT algorithm is unsupported or disallowed:

`domain_error(jwt_algorithm,Algorithm)`

`Header` or `Claims` cannot be encoded as JSON:

`domain_error(json_sink,HeaderOrClaims)`

`Key` is not a valid symmetric key:

`domain_error(jwt_symmetric_key,Key)`

`Algorithm` is not a supported signing algorithm:

`domain_error(jwt_signing_algorithm,Algorithm)`

---

sign_payload/5

Signs a payload JSON term as a compact JWS using the given header, key, and options.

Compilation flags:

`static`

Template:

`sign_payload(Header,Payload,Key,Token,Options)`

Mode and number of proofs:

`sign_payload(+term,+term,+term,-atom,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Header` does not contain a valid `alg` member:

`domain_error(jwt_header,Header)`

The JWT algorithm is unsupported or disallowed:

`domain_error(jwt_algorithm,Algorithm)`

`Header` or `Payload` cannot be encoded as JSON:

`domain_error(json_sink,HeaderOrPayload)`

`Key` is not a valid symmetric key:

`domain_error(jwt_symmetric_key,Key)`

`Algorithm` is not a supported signing algorithm:

`domain_error(jwt_signing_algorithm,Algorithm)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.174.10 `jwt_openssl`

OpenSSL-backed JWS signature verification helpers.

Availability:
`logtalk_load(jwt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-26

Compilation flags:
`static, context_switching_calls`

Uses:
crypto
jwt_der
list
os
user

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *verify/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`verify/5`

Verifies a JWS signing input and signature using OpenSSL and a PEM public key.

Compilation flags:
`static`

Template:

`verify(Algorithm,PEM,SigningInput,Signature,Options)`

Mode and number of proofs:

`verify(+atom,+atom,+atom,+list(byte),+list(compound))` - `zero_or_one_or_error`

Exceptions:

The OpenSSL executable does not exist:

`existence_error(os_command,Executable)`

`Signature` is not a 64-byte raw ES256 signature:

`domain_error(jwt_es256_signature,Signature)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.175 kcenters_clusterer

object

1.175.1 `kcenters_clusterer`

k-Centers clusterer for continuous datasets. Learns from a dataset object implementing the `clustering_dataset_protocol` protocol and returns a clusterer term that can be used for assigning new instances to clusters and exported as predicate clauses.

Availability:

`logtalk_load(kcenters_clusterer(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *clusterer_common*

Uses:

format
list
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *kmeans_clusterer*, *kmedoids_clusterer*, *kmedians_clusterer*

1.176 kemeny_young_ranker

object

1.176.1 `kemeny_young_ranker`

Kemeny-Young pairwise preference ranker. Learns one deterministic consensus order from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol by maximizing the total agreeing pairwise preference weight over all linear orders and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:
`logtalk_load(kemeny_young_ranker(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *ranking_dataset_common*

`public` *score_ranker_model_common*

`public` *condorcet_victory_common*

Uses:

avltree

list

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
  - `consensus_ranking/2`
  - `consensus_score/2`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

consensus_ranking/2

Returns the learned full-item consensus ranking selected by the Kemeny-Young optimization.

Compilation flags:
```
static
```

Template:
```
consensus_ranking(Ranker,ConsensusRanking)
```
Mode and number of proofs:
```
consensus_ranking(+compound,-list) - one
```

---

consensus_score/2

Returns the maximum total pairwise agreement score of the learned consensus ranking.

Compilation flags:
```
static
```

Template:
```
consensus_score(Ranker,ConsensusScore)
```
Mode and number of proofs:
```
consensus_score(+compound,-number) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *ranked_pairs*, *schulze_ranker*

1.177 kernel_pca_projection

object

1.177.1 `kernel_pca_projection`

Kernel Principal Component Analysis reducer for continuous datasets using a portable power-iteration eigensolver over centered kernel Gram matrices.

Availability:
`logtalk_load(kernel_pca_projection(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-08-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *dimension_reducer_common*

Uses:
format
linear_algebra
list
numberlist
pairs
type

Remarks:
(none)

Inherited public predicates:
check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pca_projection, *random_projection*

1.178 kernel_svm_classifier

object

1.178.1 `kernel_svm_classifier`

Kernel support vector machine classifier using one-vs-rest dual margin models with linear, polynomial, and radial basis function kernels.

Availability:

```
logtalk_load(kernel_svm_classifier(loader))
```

Author: Paulo Moura
Version: 1:0:2
Date: 2026-10-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public probabilistic_classifier_common
```

Uses:

format
linear_algebra
list
numberlist
pairs
type

Remarks:

(none)

Inherited public predicates:

check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *linear_svm_classifier*, *logistic_regression_classifier*, *kernel_pca_projection*

1.179 kmeans_clusterer

object

1.179.1 `kmeans_clusterer`

k-Means clusterer for continuous datasets. Learns from a dataset object implementing the `clustering_dataset_protocol` protocol and returns a clusterer term that can be used for assigning new instances to clusters and exported as predicate clauses.

Availability:

```
logtalk_load(kmeans_clusterer(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *clusterer_common*

Uses:

format
list
numberlist
type

Remarks:

(none)

Inherited public predicates:

check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *nearest_centroid_classifier*

1.180 kmedians_clusterer

object

1.180.1 `kmedians_clusterer`

k-Medians clusterer for continuous datasets. Learns from a dataset object implementing the `clustering_dataset_protocol` protocol and returns a clusterer term that can be used for assigning new instances to clusters and exported as predicate clauses.

Availability:

```
logtalk_load(kmedians_clusterer(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *clusterer_common*

Uses:

format
list
numberlist
type

Remarks:

(none)

Inherited public predicates:

check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *kmeans_clusterer*, *kmedoids_clusterer*

1.181 kmedoids_clusterer

object

1.181.1 `kmedoids_clusterer`

k-Medoids clusterer for continuous datasets. Learns from a dataset object implementing the `clustering_dataset_protocol` protocol and returns a clusterer term that can be used for assigning new instances to clusters and exported as predicate clauses.

Availability:

```
logtalk_load(kmedoids_clusterer(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *clusterer_common*

Uses:

format
list
numberlist
type

Remarks:

(none)

Inherited public predicates:

check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *kmeans_clusterer*

1.182 kmodes_clusterer

object

1.182.1 `kmodes_clusterer`

k-Modes clusterer for discrete datasets. Learns from a dataset object implementing the `clustering_dataset_protocol` protocol and returns a clusterer term that can be used for assigning new instances to clusters and exported as predicate clauses.

Availability:

```
logtalk_load(kmodes_clusterer(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *clusterer_common*

Uses:

format
list
type

Remarks:

(none)

Inherited public predicates:

check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *kprototypes_clusterer*

1.183 knn_classifier

object

1.183.1 knn_classifier

k-Nearest Neighbors classifier with multiple distance metrics and weighting options. Learns from a dataset object implementing the `dataset_protocol` protocol and returns a classifier term that can be used for prediction and exported as predicate clauses.

Availability:

```
logtalk_load(knn_classifier(loader))
```

Author: Paulo Moura
Version: 2:0:0
Date: 2026-05-11

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *probabilistic_classifier_common*

Uses:

format
list
pairs
type

Remarks:

(none)

Inherited public predicates:

check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *predict/4*
  - *predict_probabilities/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

predict/4

Predicts the class label for a new instance using the learned classifier and the given options. The instance is a list of `Attribute-Value` pairs.

Compilation flags:

```
static
```

Template:

```
predict(Classifier,Instance,Class,Options)
```

Mode and number of proofs:

```
predict(+compound,+list,-atom,+list(compound)) - one
```

---

predict_probabilities/4

Predicts class probabilities for a new instance using the learned classifier and the given options. Returns a list of `Class-Probability` pairs. The instance is a list of `Attribute-Value` pairs.

Compilation flags:

```
static
```

Template:

```
predict_probabilities(Classifier,Instance,Probabilities,Options)
```

Mode and number of proofs:

```
predict_probabilities(+compound,+list,-list,+list(compound)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *c45_classifier*, *isolation_forest_anomaly_detector*, *naive_bayes_classifier*, *nearest_centroid_classifier*, *random_forest_classifier*, *adaptive_boosting_classifier*

1.184 knn_distance_anomaly_detector

object

1.184.1 knn_distance_anomaly_detector

k-nearest-neighbor distance anomaly detector with multiple distance metrics, mixed-feature support, and missing-value handling. Learns from baseline training examples selected from a dataset object implementing the `anomaly_dataset_protocol` protocol and returns a detector term that can be used for scoring, prediction, and export.

Availability:

```
logtalk_load(knn_distance_anomaly_detector(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *anomaly_detector_common*

Uses:
format
list
numberlist
pairs
type

Remarks:
(none)

Inherited public predicates:
anomaly_detector_options/2 *check_anomaly_detector/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict/4* *print_anomaly_detector/1* *score/3* *score_all/3* *valid_anomaly_detector/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

anomaly_dataset_protocol, *anomaly_detector_protocol*, *isolation_forest_anomaly_detector*, *lof_anomaly_detector*

1.185 knn_forecasting

object

1.185.1 `knn_forecasting`

k-Nearest Neighbors (analog method) time series forecaster: matches the most recent window of observations against historical windows of the same length and predicts by aggregating what followed the most similar ones. Supports multiple distance metrics, neighbor weighting schemes, optional differencing, and missing observations represented as unbound variables.

Availability:

```
logtalk_load(knn_forecasting(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-14

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public forecaster_common
```

Uses:

format
list
numberlist
type

Remarks:

(none)

Inherited public predicates:

check_forecaster/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *forecast/3* *forecaster_options/2* *learn/2* *learn/3* *option/2* *option/3* *print_forecaster/1* *valid_forecaster/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *update/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

update/3

Returns a new forecaster after appending one observation. Observation may be left an unbound variable to represent a missing observation; see update/4.

Compilation flags:

static

Template:

update(Forecaster,Observation,UpdatedForecaster)

Mode and number of proofs:

update(+compound,@number,-compound) - one_or_error

Exceptions:

Forecaster is a variable:

instantiation_error

Forecaster is neither a variable nor a valid forecaster:

domain_error(forecaster,Forecaster)

Observation is neither a variable nor a number:

type_error(number,Observation)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

exponential_smoothing, *time_series_regression*

1.186 knn_item_recommender

object

1.186.1 knn_item_recommender

Item-based k-nearest-neighbor recommender with similarity-weighted ratings and pluggable metrics.

Availability:

```
logtalk_load(knn_item_recommender(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-04

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public recommender_common
```

Uses:

list
type

Remarks:

(none)

Inherited public predicates:

check_option/1 check_options/1 check_recommender/1 default_option/1 default_options/1 diagnostic/2 diagnostics/2 export_to_clauses/4 export_to_file/4 learn/2 learn/3 option/2 option/3 print_recommender/1 recommend/4 recommender_options/2 score/4 valid_option/1 valid_options/1 valid_recommender/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

recommender_protocol, *cosine_similarity*

1.187 knn_regression

object

1.187.1 `knn_regression`

k-Nearest Neighbors regressor with multiple distance metrics, weighting schemes, optional feature scaling, and mixed-feature support.

Availability:

```
logtalk_load(knn_regression(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public regressor_common
```

Uses:

format
list
numberlist
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

linear_regression, *regression_tree*, *random_forest_regression*, *gradient_boosting_regression*

1.188 knn_user_recommender

object

1.188.1 `knn_user_recommender`

User-based k-nearest-neighbor recommender with mean-centered predictions and pluggable similarity metrics.

Availability:
`logtalk_load(knn_user_recommender(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-04

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *recommender_common*
Uses:
list
type

Remarks:

(none)

Inherited public predicates:

check_option/1 check_options/1 check_recommender/1 default_option/1 default_options/1 diagnostic/2 diagnostics/2 export_to_clauses/4 export_to_file/4 learn/2 learn/3 option/2 option/3 print_recommender/1 recommend/4 recommender_options/2 score/4 valid_option/1 valid_options/1 valid_recommender/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

recommender_protocol, *pearson_similarity*

1.189 kprototypes_clusterer

object

1.189.1 `kprototypes_clusterer`

k-Prototypes clusterer for mixed datasets with continuous and discrete attributes. Learns from a dataset object implementing the `clustering_dataset_protocol` protocol and returns a clusterer term that can be used for assigning new instances to clusters and exported as predicate clauses.

Availability:

```
logtalk_load(kprototypes_clusterer(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *clusterer_common*

Uses:
format
list
population
type

Remarks:
(none)

Inherited public predicates:
check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *kmeans_clusterer*, *kmedoids_clusterer*

1.190 ksuid

object

1.190.1 `ksuid`

KSUID generator using atom representation and the canonical Base62 alphabet.

Availability:
`logtalk_load(ksuid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *ksuid(atom,0123456789ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz)*

Remarks:
(none)

Inherited public predicates:
generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ksuid(Representation,Alphabet), *cuid2*, *ids*, *nanoid*, *snowflakeid*, *sqids*, *typeid*, *ulid*, *uuid*

object

1.190.2 `ksuid(Representation,Alphabet)`

- `Representation` - Text representation for the KSUID. Possible values are `atom`, `chars`, and `codes`.
- `Alphabet` - Base62 alphabet used for encoding KSUIDs represented as an atom, list of characters, or list of character codes.

KSUID generator.

Availability:
`logtalk_load(ksuid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ksuid_protocol*
Uses:
crypto
iso8601
list
os

Remarks:
(none)

Inherited public predicates:
generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ksuid, *cuid2(Representation,Size,Alphabet)*, *ids(Representation,Bytes)*, *nanoid(Representation,Size,Alphabet)*, *snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node)*, *sqids(Representation)*, *typeid(Representation)*, *ulid(Representation)*, *uuid(Representation)*

protocol

1.190.3 `ksuid_protocol`

KSUID generator protocol.

Availability:
`logtalk_load(ksuid(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-26

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *generate/1*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

`generate/1`

Returns a KSUID.

Compilation flags:
`static`

Template:
`generate(KSUID)`
Mode and number of proofs:
`generate(--ground)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.191 language_detection

object

1.191.1 `af_language_profile`

Afrikaans language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_af

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.2 `ar_language_profile`

Arabic language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_ar

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.3 `bg_language_profile`

Bulgarian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_bg

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.4 `bn_language_profile`

Bengali language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_bn

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.5 `br_language_profile`

Breton language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_br

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.6 `ca_language_profile`

Catalan language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_ca

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.7 `cs_language_profile`

Czech language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_cs

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.8 `da_language_profile`

Danish language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_da

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.9 `de_language_profile`

German language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_de

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.10 `el_language_profile`

Greek language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_el

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.11 `en_language_profile`

English language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_en

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.12 `eo_language_profile`

Esperanto language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_eo

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.13 `es_language_profile`

Spanish language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_es

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.14 `et_language_profile`

Estonian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_et

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.15 `eu_language_profile`

Basque language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_eu

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.16 `fa_language_profile`

Persian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_fa

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.17 `fi_language_profile`

Finnish language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_fi

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.18 `fr_language_profile`

French language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_fr

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.19 `ga_language_profile`

Irish language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_ga

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.20 `gl_language_profile`

Galician language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_gl

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.21 `ha_language_profile`

Hausa language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_ha

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.22 `he_language_profile`

Hebrew language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_he

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.23 `hi_language_profile`

Hindi language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_hi

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.24 `hr_language_profile`

Croatian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_hr

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.25 `hu_language_profile`

Hungarian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_hu

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.26 `hy_language_profile`

Armenian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_hy

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.27 `hybrid_language_detector`

Language detection strategy combining character-trigram, stop-word, and Unicode script scores.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-07

Compilation flags:
`static, context_switching_calls`

Implements:

`public` *language_detection_strategy_protocol*

Imports:

`public` *language_detection_scoring*

Uses:

ngram_language_detector
script_language_detector
stopword_language_detector

Remarks:

(none)

Inherited public predicates:

scores/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ngram_language_detector, *stopword_language_detector*, *script_language_detector*

object

1.191.28 `id_language_profile`

Indonesian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_id

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.29 `it_language_profile`

Italian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_it

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.30 `ja_language_profile`

Japanese language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_ja

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.31 `ko_language_profile`

Korean language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_ko

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.191.32 `ku_language_profile`

Kurdish language detection profile.

Availability:

```
logtalk_load(language_detection(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *language_profile_common*

Provides:

language_profiles::default_profile/2

Uses:

stopwords_ku

Remarks:

(none)

Inherited public predicates:

language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.33 `la_language_profile`

Latin language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_la

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.191.34 `language_detection_scoring`

Shared portable language detection scoring predicates.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-15

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *cosine_similarity/3*
  - *normalize_scores/2*
  - *squared_norm/2*
  - *cosine_similarity/5*
- *Private predicates*
- *Operators*

Public predicates

(none)

Protected predicates

cosine_similarity/3

Computes cosine similarity between two lexically sorted sparse count vectors.

Compilation flags:

```
static
```

Template:

```
cosine_similarity(Counts1,Counts2,Similarity)
```

Mode and number of proofs:

```
cosine_similarity(+list(pair(atom,positive_integer)),+list(pair(atom,
positive_integer)),-float) - one
```

---

normalize_scores/2

Normalizes non-negative scores to sum to one or returns an empty list when their total is zero.

Compilation flags:

```
static
```

Template:

```
normalize_scores(RawScores,Scores)
```

Mode and number of proofs:

```
normalize_scores(+list(pair(atom,number)),-list(pair(atom,float))) - one
```

---

squared_norm/2

Computes the squared norm of a sparse count vector.

Compilation flags:

```
static
```

Template:

```
squared_norm(Counts,SquareSum)
```

Mode and number of proofs:

```
squared_norm(+list(pair(atom,positive_integer)),-float) - one
```

`cosine_similarity/5`

Computes cosine similarity using precomputed squared norms.

Compilation flags:
`static`

Template:
`cosine_similarity(Counts1,SquareSum1,Counts2,SquareSum2,Similarity)`
Mode and number of proofs:
`cosine_similarity(+list,+float,+list,+float,-float)` - `one`

Private predicates

(none)

Operators

(none)
object

1.191.35 `language_detection_scripts`

Unicode script analysis and language script registry.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-07

Compilation flags:
`static, context_switching_calls`

Uses:
list
user

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

      - *script_ratios/2*
      - *language_scripts/2*
      - *default_language_scripts/2*
      - *custom_language_scripts/2*
      - *scripts_to_languages/2*
    - *Protected predicates*
    - *Private predicates*
    - *Operators*

Public predicates

script_ratios/2

Returns observed Unicode scripts and their ratios ordered by decreasing ratio and then by script name.

Compilation flags:
```
static
```

Template:
```
script_ratios(Codes,ScriptRatios)
```
Mode and number of proofs:
```
script_ratios(+list(character_code),-list(pair(atom,float))) - one
```

---

language_scripts/2

Enumerates effective ISO 639-1 language codes and their expected Unicode scripts. Custom mappings override default mappings.

Compilation flags:
```
static
```

Template:
```
language_scripts(Language,Scripts)
```
Mode and number of proofs:
```
language_scripts(?atom,?list(atom)) - zero_or_more
```

---

default_language_scripts/2

Enumerates default ISO 639-1 language codes and their expected Unicode scripts.

Compilation flags:
```
static
```

Template:

```
default_language_scripts(Language,Scripts)
```
Mode and number of proofs:
```
default_language_scripts(?atom,?list(atom)) - zero_or_more
```

---

custom_language_scripts/2

Enumerates custom ISO 639-1 language codes and their expected Unicode scripts.

Compilation flags:
```
static, multifile
```

Template:
```
custom_language_scripts(Language,Scripts)
```
Mode and number of proofs:
```
custom_language_scripts(?atom,?list(atom)) - zero_or_more
```

---

scripts_to_languages/2

Returns the sorted unique registered languages compatible with any meaningful script in the given list.

Compilation flags:
```
static
```

Template:
```
scripts_to_languages(Scripts,Languages)
```
Mode and number of proofs:
```
scripts_to_languages(+list(atom),-list(atom)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.191.36 language_detection_strategy_protocol

Language detection strategy protocol.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *scores/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

scores/3

Returns relative detection scores for the requested ISO 639-1 candidate language codes. Assumes normalized text and validated candidates.

Compilation flags:
`static`

Template:
`scores(Codes,Candidates,Scores)`
Mode and number of proofs:
`scores(+list(character_code),+list(atom),-list(pair(atom,float)))` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

language_detector_protocol

object

1.191.37 `language_detection_tokenizer`

Language-neutral tokenizer provider for language detection.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *tokenizer_language_protocol*
Imports:
`public` *tokenizer_rules*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *split_sentence_codes/3* *tokenize_codes/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

tokenizer(Representation,Language), *tokenizer_rules*

object

1.191.38 `language_detector(Representation,Strategy)`

- `Representation` - Text representation. Valid values are `atom`, `chars`, and `codes`.
- `Strategy` - Object implementing the `language_detection_strategy_protocol` protocol.

Language detector parameterized by text representation and detection strategy.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-15

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *language_detector_protocol*
Imports:
`public` *options*
Uses:
language_profiles
list
text_normalizer(Representation,Profile)
type

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *detect/2* *detect/3* *detect/4* *detect_all/2* *detect_all/3* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

language_detector_protocol, *language_detection_strategy_protocol*

protocol

1.191.39 `language_detector_protocol`

Language detection protocol.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *detect/2*
  - *detect/3*
  - *detect/4*
  - *detect_all/2*
  - *detect_all/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

detect/2

Detects the most likely language and returns its ISO 639-1 code.

Compilation flags:

`static`

Template:

`detect(Text,Language)`

Mode and number of proofs:

`detect(+text,-atom)` - `zero_or_one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Strategy` parameter is a variable:

`instantiation_error`

The `Strategy` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Strategy)`

The `Strategy` parameter is an object identifier but not an object conforming to `language_detection_strategy_protocol`:

`domain_error(language_detection_strategy_protocol,Strategy)`

detect/3

Detects the most likely language and returns its ISO 639-1 code and relative detection score.

Compilation flags:
`static`

Template:
`detect(Text,Language,Score)`
Mode and number of proofs:
`detect(+text,-atom,-float)` - `zero_or_one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Strategy` parameter is a variable:
`instantiation_error`
The `Strategy` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Strategy)`
The `Strategy` parameter is an object identifier but not an object conforming to `language_detection_strategy_protocol`:
`domain_error(language_detection_strategy_protocol,Strategy)`

---

detect/4

Detects the most likely language using the given options and returns its ISO 639-1 code and relative detection score.

Compilation flags:
`static`

Template:
`detect(Text,Language,Score,Options)`
Mode and number of proofs:
`detect(+text,-atom,-float,+list(compound))` - `zero_or_one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Strategy` parameter is a variable:
`instantiation_error`
The `Strategy` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Strategy)`
The `Strategy` parameter is an object identifier but not an object conforming to `language_detection_strategy_protocol`:

`domain_error(language_detection_strategy_protocol,Strategy)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`detect_all/2`

Returns candidate languages ranked by descending relative detection score or an empty list when there is insufficient evidence.

Compilation flags:

`static`

Template:

`detect_all(Text,ScoredLanguages)`

Mode and number of proofs:

`detect_all(+text,-list(pair(atom,float)))` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Strategy` parameter is a variable:

`instantiation_error`

The `Strategy` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Strategy)`

The `Strategy` parameter is an object identifier but not an object conforming to `language_detection_strategy_protocol`:

`domain_error(language_detection_strategy_protocol,Strategy)`

---

detect_all/3

Returns candidate languages ranked by descending relative detection score using the given options or an empty list when there is insufficient evidence.

Compilation flags:

`static`

Template:

`detect_all(Text,ScoredLanguages,Options)`

Mode and number of proofs:

`detect_all(+text,-list(pair(atom,float)),+list(compound))` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Strategy` parameter is a variable:

`instantiation_error`

The `Strategy` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Strategy)`

The `Strategy` parameter is an object identifier but not an object conforming to `language_detection_strategy_protocol`:

`domain_error(language_detection_strategy_protocol,Strategy)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

language_detection_strategy_protocol

category

1.191.40 `language_profile_common`

Shared predicates for language detection profiles.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-15

Compilation flags:
`static`

Implements:
`public` *language_profile_protocol*
Uses:
list
n_grams(Representation)
user

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1*

- *Public predicates*
  - *trigram_counts/1*
  - *trigram_counts/2*
- *Protected predicates*

- *Private predicates*
  - *trigram_counts_/2*
- *Operators*

Public predicates

trigram_counts/1

Returns trigram counts derived from the profile stop-word inventory.

Compilation flags:
```
static
```

Template:
```
trigram_counts(Counts)
```
Mode and number of proofs:
```
trigram_counts(-list(pair(atom,positive_integer)))-one
```

---

trigram_counts/2

Returns the profile trigram counts and their squared norm, using cached values when available.

Compilation flags:
```
static
```

Template:
```
trigram_counts(Counts,SquareSum)
```
Mode and number of proofs:
```
trigram_counts(-list(pair(atom,positive_integer)),-float)-one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

trigram_counts_/2

Returns cached profile trigram counts and their squared norm when available.

Compilation flags:
```
static
```

Template:

```
trigram_counts_(Counts,SquareSum)
```

Mode and number of proofs:

```
trigram_counts_(-list(pair(atom,positive_integer)),-float) - zero_or_one
```

Operators

(none)

object

1.191.41 language_profile_expansion

Computes and stores trigram vectors and squared norms when loading language profiles.

Availability:

```
logtalk_load(language_detection(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-15

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public expanding
```

Uses:

list
n_grams(Representation)
user

Remarks:

(none)

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.191.42 `language_profile_protocol`

Language detection profile protocol.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *language/1*
  - *trigram_counts/1*
  - *stop_word/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

language/1

Returns the profile ISO 639-1 language code.

Compilation flags:
`static`

Template:
`language(Language)`
Mode and number of proofs:
`language(-atom) - one`

---

trigram_counts/1

Returns the lexically sorted character-trigram count vector.

Compilation flags:
`static`

Template:
`trigram_counts(Counts)`
Mode and number of proofs:
`trigram_counts(-list(pair(text,positive_integer))) - one`

---

stop_word/1

Enumerates canonical lowercase stop words.

Compilation flags:
`static`

Template:
`stop_word(Word)`
Mode and number of proofs:
`stop_word(?atom) - zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.191.43 `language_profiles`

Registry of language detection profiles.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *profile/2*
  - *default_profile/2*
  - *custom_profile/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

profile/2

Enumerates the effective ISO 639-1 language codes and profile objects. Custom profiles override default profiles for the same language code.

Compilation flags:

```
static
```

Template:

```
profile(Language,Profile)
```

Mode and number of proofs:

```
profile(?atom,?object_identifier) - zero_or_more
```

---

default_profile/2

Enumerates default built-in ISO 639-1 language codes and profile objects.

Compilation flags:

```
static, multifile
```

Template:

```
default_profile(Language,Profile)
```

Mode and number of proofs:

```
default_profile(?atom,?object_identifier) - zero_or_more
```

---

custom_profile/2

Enumerates custom ISO 639-1 language codes and profile objects.

Compilation flags:

```
static, multifile
```

Template:

```
custom_profile(Language,Profile)
```

Mode and number of proofs:

```
custom_profile(?atom,?object_identifier) - zero_or_more
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

language_profile_protocol

object

1.191.44 `lt_language_profile`

Lithuanian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_lt

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.45 `lv_language_profile`

Latvian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_lv

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.191.46 `mr_language_profile`

Marathi language detection profile.

Availability:

```
logtalk_load(language_detection(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *language_profile_common*

Provides:

language_profiles::default_profile/2

Uses:

stopwords_mr

Remarks:

(none)

Inherited public predicates:

language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.47 `ms_language_profile`

Malay language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_ms

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.48 `ngram_language_detector`

Language detection strategy using character-trigram cosine similarity.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-15

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *language_detection_strategy_protocol*
Imports:
`public` *language_detection_scoring*
Uses:
language_profiles
n_grams(Representation)

Remarks:
(none)

Inherited public predicates:
scores/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

language_detector(Representation,Strategy)

object

1.191.49 `nl_language_profile`

Dutch language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_nl

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.50 `no_language_profile`

Norwegian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_no

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.51 `pl_language_profile`

Polish language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_pl

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.52 `pt_language_profile`

Portuguese language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_pt

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.53 `ro_language_profile`

Romanian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_ro

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.54 `ru_language_profile`

Russian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_ru

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.55 `script_language_detector`

Language detection strategy using Unicode script evidence.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-07

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *language_detection_strategy_protocol*
Imports:
`public` *language_detection_scoring*
Uses:
language_detection_scripts
list

Remarks:
(none)

Inherited public predicates:
scores/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ngram_language_detector, *stopword_language_detector*, *hybrid_language_detector*

object

1.191.56 `sk_language_profile`

Slovak language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*
Provides:
language_profiles::default_profile/2
Uses:
stopwords_sk

Remarks:

(none)

Inherited public predicates:

language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.57 `sl_language_profile`

Slovenian language detection profile.

Availability:

```
logtalk_load(language_detection(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *language_profile_common*

Provides:

language_profiles::default_profile/2

Uses:

stopwords_sl

Remarks:

(none)

Inherited public predicates:

language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.191.58 `so_language_profile`

Somali language detection profile.

Availability:

```
logtalk_load(language_detection(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *language_profile_common*

Provides:

language_profiles::default_profile/2

Uses:

stopwords_so

Remarks:

(none)

Inherited public predicates:

language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.59 `st_language_profile`

Southern-Sotho language detection profile.

Availability:

```
logtalk_load(language_detection(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *language_profile_common*

Provides:

language_profiles::default_profile/2

Uses:

stopwords_st

Remarks:

(none)

Inherited public predicates:

language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.191.60 `stopword_language_detector`

Language detection strategy using discriminative stop-word evidence.

Availability:

`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-15

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *language_detection_strategy_protocol*

Uses:

avltree
language_profiles
list
tokenizer(Representation,Language)

Remarks:
(none)

Inherited public predicates:
scores/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

language_detector(Representation,Strategy), *language_detection_tokenizer*

object

1.191.61 `sv_language_profile`

Swedish language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_sv

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.62 `sw_language_profile`

Swahili language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_sw

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.63 `th_language_profile`

Thai language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_th

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.64 `tl_language_profile`

Tagalog language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_tl

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.65 `tr_language_profile`

Turkish language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_tr

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.66 `uk_language_profile`

Ukrainian language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_uk

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.67 `ur_language_profile`

Urdu language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_ur

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.68 `vi_language_profile`

Vietnamese language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_vi

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.69 `yo_language_profile`

Yoruba language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_yo

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.70 `zh_language_profile`

Chinese language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_zh

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.191.71 `zu_language_profile`

Zulu language detection profile.

Availability:
`logtalk_load(language_detection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *language_profile_common*

Provides:
language_profiles::default_profile/2
Uses:
stopwords_zu

Remarks:
(none)

Inherited public predicates:
language/1 *stop_word/1* *trigram_counts/1* *trigram_counts/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.192 lasso_feature_selector

object

1.192.1 `lasso_feature_selector`

Regression feature selection by maximum absolute Lasso coefficient per original feature, with optional deterministic holdout regularization search. Search requires two usable rows.

Availability:
`logtalk_load(lasso_feature_selector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *feature_selector_common*

Uses:

format
lasso_regression
list
regression_dataset_adapter(Dataset)
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_selector/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *feature_scores/2* *learn/2* *learn/3* *option/2* *option/3* *print_selector/1* *selected_features/2* *selector_options/2* *valid_option/1* *valid_options/1* *valid_selector/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *usable_examples/3*
  - *count_numeric_targets/2*
  - *check_search_count/1*
  - *group_scores/3*
  - *encoder_block/3*
  - *block_maximum/5*
  - *select_active/3*
  - *active_scores/3*
  - *model_diagnostics/8*
  - *valid_model/1*
  - *store_regressor_options/3*
  - *complete_regressor_options/2*
  - *valid_regularization_grid/1*
  - *regularization_grid_values/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

usable_examples/3

Counts known numeric targets after shared dataset validation.

Compilation flags:

```
static
```

Template:

```
usable_examples(Examples,Dataset,Count)
```

Mode and number of proofs:

```
usable_examples(+list(compound),+object_identifier,-non_negative_integer) -
one_or_error
```

Exceptions:

A known target is not numeric:

```
type_error(number,Target)
```

No known target remains:

```
domain_error(non_empty_examples,Dataset)
```

---

count_numeric_targets/2

Counts numeric targets while skipping unbound targets.

Compilation flags:

```
static
```

Template:

```
count_numeric_targets(Examples,Count)
```

Mode and number of proofs:

```
count_numeric_targets(+list(compound),-non_negative_integer) - one_or_error
```

Exceptions:

A known target is not numeric:

```
type_error(number,Target)
```

---

check_search_count/1

Requires at least two known numeric targets for holdout search.

Compilation flags:

```
static
```

Template:

```
check_search_count(Count)
```

Mode and number of proofs:

```
check_search_count(+non_negative_integer) - one_or_error
```

Exceptions:

Fewer than two usable rows remain:

```
domain_error(lasso_search_examples,Count)
```

---

group_scores/3

Consumes exact encoder blocks and returns original feature scores in encoder order.

Compilation flags:

```
static
```

Template:

```
group_scores(Encoders,Weights,Scores)
```

Mode and number of proofs:

```
group_scores(+list(compound),+list(float),-list(pair)) - zero_or_one
```

---

encoder_block/3

Returns an encoder name and its number of coefficient columns.

Compilation flags:

```
static
```

Template:

```
encoder_block(Encoder,Feature,Count)
```

Mode and number of proofs:

```
encoder_block(+compound,-atom,-positive_integer) - zero_or_one
```

---

block_maximum/5

Consumes a coefficient block and computes its maximum absolute weight.

Compilation flags:
`static`

Template:
`block_maximum(Count,Weights,Maximum0,Maximum,Remaining)`
Mode and number of proofs:
`block_maximum(+non_negative_integer,+list(float),+float,-float,-list(float))` - `zero_or_one`

---

select_active/3

Applies the selection strategy only to groups strictly above the coefficient cutoff.

Compilation flags:
`static`

Template:
`select_active(Scores,Options,Selected)`
Mode and number of proofs:
`select_active(+list(pair),+list(compound),-list(atom))` - `one`

---

active_scores/3

Retains scores strictly above the coefficient cutoff.

Compilation flags:
`static`

Template:
`active_scores(Scores,Cutoff,Active)`
Mode and number of proofs:
`active_scores(+list(pair),+number,-list(pair))` - `one`

---

model_diagnostics/8

Builds counts, effective options, aggregation metadata, and complete nested learner diagnostics.

Compilation flags:

```
static
```

Template:

```
model_diagnostics(Regressor,Scores,Selected,OriginalCount,UsableCount,Options,
SearchResult,Diagnostics)
```

Mode and number of proofs:

```
model_diagnostics(+compound,+list(pair),+list(atom),+positive_integer,
+positive_integer,+list(compound),+term,-list(compound)) - one
```

---

valid_model/1

Checks the retained regressor and recomputes scores, selection, and metadata without binding a partial input.

Compilation flags:

```
static
```

Template:

```
valid_model(Selector)
```

Mode and number of proofs:

```
valid_model(+ground) - zero_or_one
```

Exceptions:

The retained regressor is invalid:

```
domain_error(regressor,Regressor)
```

A stored option is invalid:

```
domain_error(option,Option)
```

A generated regularization bound exceeds backend floating-point limits:

```
evaluation_error(float_overflow)
```

Stored options are not a list:

```
type_error(list,Options)
```

A stored option is not compound:

```
type_error(compound,Option)
```

---

store_regressor_options/3

Replaces the first nested options occurrence with the learner effective options, preserving later occurrences.

Compilation flags:
```
static
```

Template:
```
store_regressor_options(Options,RegressorOptions,Stored)
```
Mode and number of proofs:
```
store_regressor_options(+list(compound),+list(compound),-list(compound)) - one
```

---

complete_regressor_options/2

Checks that each learner default option indicator has an effective value.

Compilation flags:
```
static
```

Template:
```
complete_regressor_options(Defaults,Options)
```
Mode and number of proofs:
```
complete_regressor_options(+list(compound),+list(compound)) - zero_or_one
```

---

valid_regularization_grid/1

Checks an explicit penalty list or an increasing bounded linear grid with at least two candidates.

Compilation flags:
```
static
```

Template:
```
valid_regularization_grid(Grid)
```
Mode and number of proofs:
```
valid_regularization_grid(@term) - zero_or_one
```

`regularization_grid_values/2`

Expands a validated linear grid to floating-point penalties or preserves an explicit penalty list.

Compilation flags:
`static`

Template:
`regularization_grid_values(Grid,Values)`
Mode and number of proofs:
`regularization_grid_values(+term,-list(number))` - `one_or_error`

Exceptions:
A generated bound exceeds backend floating-point limits:
`evaluation_error(float_overflow)`

Operators

(none)
object

1.192.2 `regression_dataset_adapter(Dataset)`

- `Dataset` - Feature dataset object.

Transient regression view of a feature dataset, preserving missing features and excluding unknown targets.

Availability:
`logtalk_load(lasso_feature_selector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *regression_dataset_protocol*
Uses:
list
type

Remarks:
(none)

Inherited public predicates:
attribute_values/2 *example/3* *target/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *check_declarations/1*
  - *distinct_values/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

check_declarations/1

Checks distinct atom feature names and continuous or non-empty, distinct atomic categorical domains.

Compilation flags:

```
static
```

Template:

```
check_declarations(Declarations)
```

Mode and number of proofs:

```
check_declarations(+list(pair)) - one_or_error
```

Exceptions:

A feature declaration is unsupported:

```
domain_error(feature_type,Feature-Values)
```

A feature is declared more than once:

```
domain_error(duplicate_feature,Feature)
```

---

distinct_values/1

True when categorical values are pairwise distinct.

Compilation flags:

```
static
```

Template:

```
distinct_values(Values)
```

Mode and number of proofs:

```
distinct_values(+list(atomic)) - zero_or_one
```

Operators

(none)
object

1.192.3 regression_examples_adapter(Declarations,Examples)

- `Declarations` - Validated feature declarations.
- `Examples` - Validated feature dataset examples.

Transient materialized regression subset with validated declarations and numeric targets.

Availability:
`logtalk_load(lasso_feature_selector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *regression_dataset_protocol*
Uses:
list

Remarks:
(none)

Inherited public predicates:
attribute_values/2 *example/3* *target/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.193 lasso_regression

object

1.193.1 `lasso_regression`

Lasso regression regressor supporting continuous and mixed-feature datasets using coordinate descent with coefficient-wise L1 shrinkage. Learns from a dataset object implementing the `regression_dataset_protocol` protocol and returns a regressor term that can be used for prediction and exported as predicate clauses.

Availability:
`logtalk_load(lasso_regression(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *regressor_common*
Uses:
format
list
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

linear_regression, *ridge_regression*, *knn_regression*, *regression_tree*, *random_forest_regression*, *gradient_boosting_regression*

1.194 lda_classifier

object

1.194.1 `lda_classifier`

Linear Discriminant Analysis classifier for continuous datasets using a pooled regularized covariance model.

Availability:
`logtalk_load(lda_classifier(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *classifier_common*
Uses:
format
linear_algebra

list
pairs
type

Remarks:
(none)

Inherited public predicates:
check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *predict_scores/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

predict_scores/3

Predicts class discriminant scores for a new instance.

Compilation flags:
`static`

Template:
`predict_scores(Classifier,Instance,Scores)`
Mode and number of proofs:
`predict_scores(+compound,+list,-list)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *qda_classifier*, *nearest_centroid_classifier*, *lda_projection*

1.195 lda_projection

object

1.195.1 `lda_projection`

Linear Discriminant Analysis projection for labeled continuous datasets using a portable regularized Fisher eigensolver.

Availability:

```
logtalk_load(lda_projection(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public dimension_reducer_common
```

Uses:

format
linear_algebra
list
pairs
type

Remarks:

(none)

Inherited public predicates:

check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pca_projection, *random_projection*

1.196 lemmatization

object

1.196.1 `english_lemmatizer`

English lemmatization provider using curated exceptions, base forms, and conservative inflection rules.

Availability:
`logtalk_load(lemmatization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-05

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *lemmatizer_language_protocol*
Uses:
list

Remarks:
(none)

Inherited public predicates:
lemma/2 *lemma/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

lemmatizer(Representation,Language)

object

1.196.2 `lemmatizer(Representation,Language)`

- `Representation` - Word representation. Valid values are `atom`, `codes`, and `chars`.
- `Language` - Object implementing the `lemmatizer_language_protocol` protocol.

Lemmatizer parameterized by text representation and language provider.

Availability:
`logtalk_load(lemmatization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *lemmatizer_protocol*
Imports:
`public` *options*
Uses:
list
string(Representation)

type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *lemma/2* *lemma/3* *lemmas/2* *lemmas/3* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

lemmatizer_protocol, *lemmatizer_language_protocol*

protocol

1.196.3 `lemmatizer_language_protocol`

Language provider protocol for lemmatization using canonical lowercase atoms.

Availability:
`logtalk_load(lemmatization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *lemma/2*
  - *lemma/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

lemma/2

Enumerates distinct recognized lemmas in stable preference order. Fails for an unknown word.

Compilation flags:
`static`

Template:
`lemma(Word,Lemma)`
Mode and number of proofs:
`lemma(+atom,-atom) - zero_or_more`

---

lemma/3

Enumerates distinct recognized lemmas for a part-of-speech hint in stable preference order. Fails for an unknown word or unsupported hint.

Compilation flags:
`static`

Template:
`lemma(Word,PartOfSpeech,Lemma)`
Mode and number of proofs:
`lemma(+atom,+atom,-atom) - zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

lemmatizer_protocol, *lemmatizer(Representation,Language)*

protocol

1.196.4 `lemmatizer_protocol`

Lemmatization protocol for words represented as atoms, lists of characters, or lists of character codes.

Availability:
`logtalk_load(lemmatization(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - `lemma/2`
  - `lemma/3`
  - `lemmas/2`
  - `lemmas/3`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

lemma/2

Enumerates the distinct lemmas of a word in provider preference order. Returns the normalized word when no lemma is recognized.

Compilation flags:

`static`

Template:

`lemma(Word,Lemma)`

Mode and number of proofs:

`lemma(+text,-text)` - `one_or_more_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Language` parameter is a variable:

`instantiation_error`

The `Language` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Language)`

The `Language` parameter is an object identifier but not an object conforming to `lemmatizer_language_protocol`:

`domain_error(lemmatizer_language_protocol,Language)`

`Word` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Word` is not an atom:

`type_error(atom,Word)`

The `Representation` parameter is `chars` but `Word` is not a list of characters:

`type_error(chars,Word)`

The `Representation` parameter is `codes` but `Word` is not a list of character codes:

`type_error(codes,Word)`

---

lemma/3

Lemmatizes a word using the given options. Recognized options are `part_of_speech(PartOfSpeech)`, `ambiguity(Policy)`, where `Policy` is `first` or `all`, and `unknown(Policy)`, where `Policy` is `normalize`, `preserve`, or `fail`.

Compilation flags:

`static`

Template:

`lemma(Word,Lemma,Options)`

Mode and number of proofs:

`lemma(+text,-text,+list(compound))` - `zero_or_more_or_error`

Exceptions:

The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Language` parameter is a variable:
`instantiation_error`
The `Language` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Language)`
The `Language` parameter is an object identifier but not an object conforming to `lemmatizer_language_protocol`:
`domain_error(lemmatizer_language_protocol,Language)`
`Word` is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but `Word` is not an atom:
`type_error(atom,Word)`
The `Representation` parameter is `chars` but `Word` is not a list of characters:
`type_error(chars,Word)`
The `Representation` parameter is `codes` but `Word` is not a list of character codes:
`type_error(codes,Word)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
An element of the list `Options` is a `parts_of_speech/1` option, which is only valid for `lemmas/3`:
`domain_error(option,parts_of_speech(A))`

`lemmas/2`

Lemmatizes a list of words, selecting the first lemma for each word and preserving order. Returns normalized words when no lemma is recognized.

Compilation flags:
`static`

Template:
`lemmas(Words,Lemmas)`
Mode and number of proofs:
`lemmas(+list(text),-list(text))` - `one_or_error`

Exceptions:

The Representation parameter is a variable:
instantiation_error
The Representation parameter is neither a variable nor atom, chars, or codes:
domain_error(text_representation,Representation)
The Language parameter is a variable:
instantiation_error
The Language parameter neither a variable nor an object identifier:
type_error(object_identifier,Language)
The Language parameter is an object identifier but not an object conforming to lemmatizer_language_protocol:
domain_error(lemmatizer_language_protocol,Language)
Words is a variable or a partial list:
instantiation_error
Words is neither a variable nor a list:
type_error(list,Words)
An element Word of the list Words is not a ground term:
instantiation_error
The Representation parameter is atom but a ground element Word of the list Words is not an atom:
type_error(atom,Word)
The Representation parameter is chars but an element Word of the list Words is not a list of characters:
type_error(chars,Word)
The Representation parameter is codes but an element Word of the list Words is not a list of character codes:
type_error(codes,Word)

---

lemmas/3

Lemmatizes a list of words using the given options. In addition to the scalar options, accepts parts_of_speech(PartsOfSpeech) for per-word hints. The part_of_speech/1 and parts_of_speech/1 options are mutually exclusive.

Compilation flags:
static

Template:
lemmas(Words,Lemmas,Options)
Mode and number of proofs:
lemmas(+list(text),-list(text),+list(compound)) - zero_or_more_or_error

Exceptions:
The Representation parameter is a variable:
instantiation_error
The Representation parameter is neither a variable nor atom, chars, or codes:
domain_error(text_representation,Representation)
The Language parameter is a variable:
instantiation_error
The Language parameter neither a variable nor an object identifier:
type_error(object_identifier,Language)

The `Language` parameter is an object identifier but not an object conforming to `lemmatizer_language_protocol`:
```
domain_error(lemmatizer_language_protocol,Language)
```
`Words` is a variable or a partial list:
```
instantiation_error
```
`Words` is neither a variable nor a list:
```
type_error(list,Words)
```
An element `Word` of the list `Words` is not a ground term:
```
instantiation_error
```
The `Representation` parameter is `atom` but a ground element `Word` of the list `Words` is not an atom:
```
type_error(atom,Word)
```
The `Representation` parameter is `chars` but an element `Word` of the list `Words` is not a list of characters:
```
type_error(chars,Word)
```
The `Representation` parameter is `codes` but an element `Word` of the list `Words` is not a list of character codes:
```
type_error(codes,Word)
```
`Options` is a variable:
```
instantiation_error
```
`Options` is neither a variable nor a list:
```
type_error(list,Options)
```
An element `Option` of the list `Options` is a variable:
```
instantiation_error
```
An element `Option` of the list `Options` is neither a variable nor a compound term:
```
type_error(compound,Option)
```
An element `Option` of the list `Options` is a compound term but not a valid option:
```
domain_error(option,Option)
```
The `part_of_speech(POS)` and `parts_of_speech(PsOS)` options are both present:
```
consistency_error(mutually_exclusive_options,part_of_speech(POS),
parts_of_speech(PsOS))
```
The `Words` and `PartsOfSpeech` lists have different lengths:
```
consistency_error(same_length,Words,PartsOfSpeech)
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

lemmatizer_language_protocol, *lemmatizer(Representation,Language)*

1.197 lgtdoc

object

1.197.1 `lgtdoc`

Documenting tool. Generates XML documenting files for loaded entities and for library, directory, entity, and predicate indexes.

Availability:
`logtalk_load(lgtdoc(loader))`

Author: Paulo Moura
Version: 11:6:0
Date: 2026-10-27

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *lgtdocp*
Imports:
`public` *tool_diagnostics_common*
`public` *options*
Uses:
date
list
logtalk
os
term_io
type
user
varlist

Remarks:
(none)

Inherited public predicates:
all/0 all/1 check_option/1 check_options/1 context_summaries/2 default_option/1 default_options/1 diagnostic/2 diagnostic/3 diagnostic_rule/5 diagnostic_rules/1 diagnostic_target/1 diagnostics/2 diagnostics/3 diagnostics_breakdown/2 diagnostics_preflight/2 diagnostics_preflight/3 diagnostics_summary/2 diagnostics_summary/3 diagnostics_tool/5 directories/1 directories/2 directory/1 directory/2 file/1 file/2 files/1 files/2 fix_option/2 fix_options/2 libraries/1 libraries/2 library/1 library/2 merge_options/2 option/2 option/3 rdirectories/1 rdirectories/2 rdirectory/1 rdirectory/2 rlibraries/1 rlibraries/2 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *library_entity_/4*

      - *directory_entity_/4*
      - *type_entity_/4*
      - *predicate_entity_/4*
      - *cached_run_/5*
      - *active_diagnostic_/1*
      - *active_preflight_issue_/1*
      - *active_collection_mode_/1*
  - *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

library_entity_/4

Table of documented entities per library.

Compilation flags:
`dynamic`

Template:
`library_entity_(Library,PrimarySortKey,SecondarySortKey,Entity)`
Mode and number of proofs:
`library_entity_(?atom,?nonvar,?nonvar,?atom)` - `zero_or_more`

---

directory_entity_/4

Table of documented entities per directory.

Compilation flags:
`dynamic`

Template:
`directory_entity_(Directory,PrimarySortKey,SecondarySortKey,Entity)`
Mode and number of proofs:
`directory_entity_(?atom,?nonvar,?nonvar,?atom)` - `zero_or_more`

---

type_entity_/4

Table of documented entities per type.

Compilation flags:
```
dynamic
```

Template:
```
type_entity_(Type,PrimarySortKey,SecondarySortKey,Entity)
```
Mode and number of proofs:
```
type_entity_(?atom,?nonvar,?nonvar,?atom) - zero_or_more
```

---

predicate_entity_/4

Table of public predicates for all documented entities.

Compilation flags:
```
dynamic
```

Template:
```
predicate_entity_(Predicate,PrimarySortKey,SecondarySortKey,Entity)
```
Mode and number of proofs:
```
predicate_entity_(?predicate_indicator,?nonvar,?nonvar,?entity_identifier) -
zero_or_more
```

---

cached_run_/5

Cache of diagnostics collected while generating documentation for a target and merged options.

Compilation flags:
```
dynamic
```

Template:
```
cached_run_(Target,Options,Contexts,Diagnostics,PreflightIssues)
```
Mode and number of proofs:
```
cached_run_(?nonvar,?list(compound),?list(compound),?list(compound),?
list(compound)) - zero_or_more
```

---

active_diagnostic_/1

Diagnostics collected during the current documentation run.

Compilation flags:
`dynamic`

Template:
`active_diagnostic_(Diagnostic)`
Mode and number of proofs:
`active_diagnostic_(?compound)` - `zero_or_more`

---

active_preflight_issue_/1

Preflight issues collected during the current documentation run.

Compilation flags:
`dynamic`

Template:
`active_preflight_issue_(Issue)`
Mode and number of proofs:
`active_preflight_issue_(?compound)` - `zero_or_more`

---

active_collection_mode_/1

Current diagnostics collection mode.

Compilation flags:
`dynamic`

Template:
`active_collection_mode_(Mode)`
Mode and number of proofs:
`active_collection_mode_(?atom)` - `zero_or_one`

Operators

(none)
category

1.197.2 lgtdoc_messages

Logtalk documentation tool default message translations.

Availability:
`logtalk_load(lgtdoc(loader))`

Author: Paulo Moura
Version: 4:0:1
Date: 2024-12-02

Compilation flags:
`static`

Provides:
logtalk::message_prefix_stream/4
logtalk::message_tokens//2

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.197.3 lgtdocp

Documenting tool protocol.

Availability:
`logtalk_load(lgtdoc(loader))`

Author: Paulo Moura
Version: 6:0:0
Date: 2024-03-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- Compiling files for generating XML documentation: All source files must be compiled with the `source_data` flag turned on.
- `xml_spec(Specification)` option: XML documenting files specification format. Possible option values are `dtd` (DTD specification; default) and `xsd` (XML Schema specification).
- `xml_spec_reference(Reference)` option: Reference to the XML specification file in XML documenting files. Possible values are `local` (default; DTD/XSD file in same folder as XML files), `web` (logtalk.org website DTD/XSD file), and `standalone` (no reference to specification files).
- `entity_xsl_file(File)` option: XSLT file to use with generated XML documenting files. Default is `logtalk_entity_to_xml.xsl`, allowing the XML files to be viewed by opening them with a browser supporting XSLT (after running the `lgt2xml.sh` script on the output directory).
- `index_xsl_file(File)` option: XSLT file to use with generated XML documenting files. Default is `logtalk_index_to_xml.xsl`, allowing the XML files to be viewed by opening them with a browser supporting XSLT (after running the `lgt2xml.sh` script on the output directory).
- `xml_docs_directory(Directory)` option: Directory where the XML documenting files will be generated. The default value is `./xml_docs`, a sub-directory of the source files directory.
- `bom(Boolean)` option: Defines if a BOM should be added to the generated XML documenting files.
- `encoding(Encoding)` option: Encoding to be used for the generated XML documenting files.
- `omit_path_prefixes(Prefixes)` option: List of path prefixes (atoms) to omit when writing directory paths. The default value is to omit the home directory.
- `exclude_files(List)` option: List of files to exclude when generating the XML documenting files.
- `exclude_paths(List)` option: List of relative library paths to exclude when generating the XML documenting files (default is `[]`). All sub-directories of the excluded directories are also excluded.
- `exclude_prefixes(List)` option: List of path prefixes to exclude when generating the XML documenting files (default is `[]`).

- `exclude_entities(List)` option: List of entities to exclude when generating the XML documenting files (default is `[]`).
- `sort_predicates(Boolean)` option: Sort entity predicates (default is `false`).
- Known issues: Some options may depend on the used XSL processor. Most XSL processors support DTDs but only some of them support XML Schemas. Some processors (e.g., `fop2`) reject reference to a DTD.

Inherited public predicates:

(none)

- *Public predicates*
  - *rlibraries/2*
  - *rlibraries/1*
  - *rlibrary/2*
  - *rlibrary/1*
  - *libraries/2*
  - *libraries/1*
  - *library/2*
  - *library/1*
  - *rdirectories/2*
  - *rdirectories/1*
  - *rdirectory/2*
  - *rdirectory/1*
  - *directories/2*
  - *directories/1*
  - *directory/2*
  - *directory/1*
  - *files/2*
  - *files/1*
  - *file/2*
  - *file/1*
  - *all/1*
  - *all/0*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

rlibraries/2

Creates XML documenting files for all entities in all given libraries and their sub-libraries using the specified options.

Compilation flags:

```
static
```

Template:

```
rlibraries(Libraries,Options)
```

Mode and number of proofs:

```
rlibraries(+list(atom),+list) - one
```

---

rlibraries/1

Creates XML documenting files for all entities in all given libraries and their sub-libraries using default options.

Compilation flags:

```
static
```

Template:

```
rlibraries(Libraries)
```

Mode and number of proofs:

```
rlibraries(+list(atom)) - one
```

---

rlibrary/2

Creates XML documenting files for all entities in a library and its sub-libraries using the specified options.

Compilation flags:

```
static
```

Template:

```
rlibrary(Library,Options)
```

Mode and number of proofs:

```
rlibrary(+atom,+list) - one
```

Examples:

Generate XML documenting files for all tool entities for later conversion to Markdown files

```
rlibrary(tools,[xslfile('lgtmd.xsl')])
yes
```

---

rlibrary/1

Creates XML documenting files for all entities in a library and its sub-libraries using default options.

Compilation flags:

```
static
```

Template:

```
rlibrary(Library)
```

Mode and number of proofs:

```
rlibrary(+atom) - one
```

Examples:

Generate XML documenting files for all tool entities for direct viewing in a browser (after indexing using the `lgt2xml` script)

```
rlibrary(tools)
yes
```

---

libraries/2

Creates XML documenting files for all entities in all given libraries using the specified options.

Compilation flags:

```
static
```

Template:

```
libraries(Libraries,Options)
```

Mode and number of proofs:

```
libraries(+list(atom),+list) - one
```

---

libraries/1

Creates XML documenting files for all entities in all given libraries using default options.

Compilation flags:

```
static
```

Template:

```
libraries(Libraries)
```

Mode and number of proofs:

```
libraries(+list(atom)) - one
```

---

library/2

Creates XML documenting files for all entities in a library using the specified options.

Compilation flags:

```
static
```

Template:

```
library(Library,Options)
```

Mode and number of proofs:

```
library(+atom,+list) - one
```

Examples:

Generate XML documenting files for all library entities for later conversion to PDF A4 files

```
library(library,[xslfile('logtalk_entity_to_pdf_a4.xsl')])
yes
```

---

library/1

Creates XML documenting files for all entities in a library using default options.

Compilation flags:

```
static
```

Template:

```
library(Library)
```

Mode and number of proofs:

```
library(+atom) - one
```

---

rdirectories/2

Creates XML documenting files for all entities in all given directories and their sub-directories using the specified options.

Compilation flags:

```
static
```

Template:

```
rdirectories(Directories,Options)
```

Mode and number of proofs:

```
rdirectories(+list(atom),+list) - one
```

---

rdirectories/1

Creates XML documenting files for all entities in all given directories and their sub-directories using default options.

Compilation flags:

```
static
```

Template:

```
rdirectories(Directories)
```

Mode and number of proofs:

```
rdirectories(+list(atom)) - one
```

---

rdirectory/2

Creates XML documenting files for all entities in a directory and its sub-directories using the specified options.

Compilation flags:
```
static
```

Template:
```
rdirectory(Directory,Options)
```
Mode and number of proofs:
```
rdirectory(+atom,+list) - one
```

Examples:

Generate XML documenting files for all entities in the tools directory for later conversion to Markdown files
```
rdirectory('./tools',[xslfile('lgtmd.xsl')])
yes
```

---

rdirectory/1

Creates XML documenting files for all entities in a directory and its sub-directories using default options.

Compilation flags:
```
static
```

Template:
```
rdirectory(Directory)
```
Mode and number of proofs:
```
rdirectory(+atom) - one
```

Examples:

Generate XML documenting files for all entities in the tools directory for direct viewing in a browser (after indexing using the `lgt2xml` script)
```
rdirectory('./tools')
yes
```

---

directories/2

Creates XML documenting files for all entities in all given directories using the specified options.

Compilation flags:
```
static
```

Template:
```
directories(Directories,Options)
```
Mode and number of proofs:

```
directories(+list(atom),+list) - one
```

---

directories/1

Creates XML documenting files for all entities in all given directories using default options.

Compilation flags:

```
static
```

Template:

```
directories(Directories)
```

Mode and number of proofs:

```
directories(+list(atom)) - one
```

---

directory/2

Creates XML documenting files for all entities in a directory using the specified options.

Compilation flags:

```
static
```

Template:

```
directory(Directory,Options)
```

Mode and number of proofs:

```
directory(+atom,+list) - one
```

Examples:

Generate XML documenting files for all the entities in the current directory for later conversion to PDF A4 files

```
directory('.',[xslfile('logtalk_entity_to_pdf_a4.xsl')])
yes
```

---

directory/1

Creates XML documenting files for all entities in a directory using default options.

Compilation flags:

```
static
```

Template:

```
directory(Directory)
```

Mode and number of proofs:

```
directory(+atom) - one
```

---

files/2

Creates XML documenting files for all entities in loaded source files using the specified options. The files can be given by name, basename, full path, or using library notation.

Compilation flags:

```
static
```

Template:

```
files(Files,Options)
```

Mode and number of proofs:

```
files(+list(atom),+list) - one
```

---

files/1

Creates XML documenting files for all entities in loaded source files using default options. The files can be given by name, basename, full path, or using library notation.

Compilation flags:

```
static
```

Template:

```
files(Files)
```

Mode and number of proofs:

```
files(+list(atom)) - one
```

---

file/2

Creates XML documenting files for all entities in a loaded source file using the specified options. The file can be given by name, basename, full path, or using library notation.

Compilation flags:

```
static
```

Template:

```
file(File,Options)
```

Mode and number of proofs:

```
file(+atom,+list) - one
```

---

file/1

Creates XML documenting files for all entities in a loaded source file using default options. The file can be given by name, basename, full path, or using library notation.

Compilation flags:
`static`

Template:
`file(File)`
Mode and number of proofs:
`file(+atom)` - `one`

---

all/1

Creates XML documenting files for all loaded entities using the specified options.

Compilation flags:
`static`

Template:
`all(Options)`
Mode and number of proofs:
`all(+list)` - `one`

---

all/0

Creates XML documenting files for all loaded entities using default options.

Compilation flags:
`static`

Mode and number of proofs:
`all` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

lgtdoc

1.198 lgtunit

object

1.198.1 `automation_report`

Intercepts unit test execution messages and generates a `*.totals` files for parsing by the `logtalk_tester.sh` automation shell script.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 6:0:1
Date: 2026-01-25

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
user

Remarks:

- Usage: Automatically loaded by the `logtalk_tester.sh` shell script.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.198.2 `cobertura_report`

Intercepts unit test execution messages and generates a `cobertura.xml` file with code coverage results.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-18

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
logtalk

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(cobertura_report))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *timestamp_/6*
  - *class_/5*
  - *method_/8*

- *lines_covered_/1*
- *lines_total_/1*
- *summary_/2*

- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

timestamp_/6

Test start date and time.

Compilation flags:
`dynamic`

Template:
`timestamp_(Year,Month,Day,Hours,Minutes,Seconds)`
Mode and number of proofs:
`timestamp_(?integer,?integer,?integer,?integer,?integer,?integer)` - `zero_or_one`

---

class_/5

Table of entity coverage data.

Compilation flags:
`dynamic`

Template:
`class_(Entity,File,Line,Covered,Total)`
Mode and number of proofs:
`class_(?entity_identifier,?atom,?integer,?integer,?integer)` - `zero_or_more`

---

method_/8

Table of predicate coverage data.

Compilation flags:
```
dynamic
```

Template:
```
method_(Entity,File,Predicate,Line,Covered,Total,Percentage,Clauses)
```
Mode and number of proofs:
```
method_(?entity_identifier,?atom,?predicate_indicator,?integer,?integer,?integer,
?float,?list) - zero_or_more
```

---

lines_covered_/1

Counter for total covered lines.

Compilation flags:
```
dynamic
```

Template:
```
lines_covered_(Covered)
```
Mode and number of proofs:
```
lines_covered_(?integer) - zero_or_one
```

---

lines_total_/1

Counter for total lines.

Compilation flags:
```
dynamic
```

Template:
```
lines_total_(Total)
```
Mode and number of proofs:
```
lines_total_(?integer) - zero_or_one
```

---

summary_/2

Coverage summary with covered and total entity counts.

Compilation flags:
`dynamic`

Template:
`summary_(Covered,Total)`
Mode and number of proofs:
`summary_(?integer,?integer)` - `zero_or_one`

Operators

(none)
object

1.198.3 coverage_report

Intercepts unit test execution messages and generates a `coverage_report.xml` file with a test suite code coverage results.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 3:2:2
Date: 2026-10-18

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
logtalk
user

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(coverage_report))`.

Inherited public predicates:
(none)

- *Public predicates*

- *Protected predicates*
- *Private predicates*
  - *timestamp_/6*
  - *object_file_/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

timestamp_/6

Cache of the starting tests timestamp.

Compilation flags:
`dynamic`

Template:
`timestamp_(Year,Month,Day,Hours,Minutes,Seconds)`
Mode and number of proofs:
`timestamp_(-integer,-integer,-integer,-integer,-integer,-integer)` - `one`

---

object_file_/2

Cache of test object - file pairs.

Compilation flags:
`dynamic`

Template:
`object_file_(Object,File)`
Mode and number of proofs:
`object_file_(?object_identifier,?atom)` - `zero_or_more`

Operators

(none)
object

1.198.4 ctrf_output

Intercepts unit test execution messages and outputs a report using the CTRF JSON format to the current output stream.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-27

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(ctrf_output))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *message_cache_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

message_cache_/1

Table of messages emitted by the lgtunit tool when running tests.

Compilation flags:
`dynamic`

Template:
`message_cache_(Message)`
Mode and number of proofs:
`message_cache_(?callable)` - `zero_or_more`

Operators

(none)
object

1.198.5 ctrf_report

Intercepts unit test execution messages and generates a `ctrf_report.json` file using the CTRF JSON format in the same directory as the tests object file.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-27

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
logtalk

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(ctrf_report))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *message_cache_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

message_cache_/1

Table of messages emitted by the lgtunit tool when running tests.

Compilation flags:
`dynamic`

Template:
`message_cache_(Message)`
Mode and number of proofs:
`message_cache_(?callable)` - `zero_or_more`

Operators

(none)
object

1.198.6 lcov_report

Intercepts unit test execution messages and generates a `lcov.info` file with code coverage results.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-18

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
logtalk

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(lcov_report))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *entity_file_/2*
  - *predicate_coverage_/8*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

entity_file_/2

Table of entity source files.

Compilation flags:
`dynamic`

Template:
`entity_file_(Entity,File)`
Mode and number of proofs:
`entity_file_(?entity_identifier,?atom)` - `zero_or_more`

predicate_coverage_/8

Table of predicate coverage data.

Compilation flags:
`dynamic`

Template:
`predicate_coverage_(Entity,File,Predicate,Line,Covered,Total,Percentage,Clauses)`
Mode and number of proofs:
`predicate_coverage_(?entity_identifier,?atom,?predicate_indicator,?integer,?integer,?integer,?float,?list)` - `zero_or_more`

Operators

(none)
object

1.198.7 lgtunit

A unit test framework supporting predicate clause coverage, determinism testing, input/output testing, property-based testing, and multiple test dialects.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 22:12:0
Date: 2026-11-17

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Imports:
`public` *tool_diagnostics_common*
`public` *lgtunit_messages*
`public` *tutor_explanations*
`public` *options*
Provides:
logtalk::message_hook/4
logtalk::trace_event/2
Uses:
fast_random
list
logtalk
numberlist

os
term_io
type
user

Remarks:

- Usage: Define test objects as extensions of the `lgtunit` object and compile their source files using the compiler option `hook(lgtunit)`.
- Portability: Deterministic unit tests are currently not available when using Quintus Prolog as the backend compiler.
- Known issues: Parameter variables cannot currently be used in the definition of test options.

Inherited public predicates:

check_option/1 check_options/1 default_option/1 default_options/1 diagnostic/2 diagnostic/3 diagnostic_rule/5 diagnostic_rules/1 diagnostic_target/1 diagnostics/2 diagnostics/3 diagnostics_preflight/2 diagnostics_preflight/3 diagnostics_summary/2 diagnostics_summary/3 diagnostics_tool/5 explain//1 failed_test_reason//1 goal_expansion/2 option/2 option/3 term_expansion/2 valid_option/1 valid_options/1

- *Public predicates*
  - *cover/1*
  - *run/0*
  - *run/1*
  - *run/2*
  - *run_test_sets/1*
  - *test/1*
  - *number_of_tests/1*
  - *deterministic/1*
  - *deterministic/2*
  - *assertion/1*
  - *assertion/2*
  - *quick_check/3*
  - *quick_check/2*
  - *quick_check/1*
  - *benchmark/2*
  - *benchmark_reified/3*
  - *benchmark/3*
  - *benchmark/4*
  - *variant/2*
  - *approximately_equal/2*
  - *approximately_equal/3*
  - *essentially_equal/3*
  - *tolerance_equal/4*
  - *=~= / 2*
  - *epsilon/1*
- *Protected predicates*
  - *run_tests/0*
  - *run_tests/1*
  - *run_test_set/0*

```
      - run_quick_check_tests/5
      - condition/0
      - setup/0
      - cleanup/0
      - make/1
      - note/1
      - file_path/2
      - file_url/2
      - suppress_text_output/0
      - suppress_binary_output/0
      - set_text_input/3
      - set_text_input/2
      - set_text_input/1
      - check_text_input/2
      - check_text_input/1
      - text_input_assertion/3
      - text_input_assertion/2
      - clean_text_input/0
      - set_binary_input/3
      - set_binary_input/2
      - set_binary_input/1
      - check_binary_input/2
      - check_binary_input/1
      - binary_input_assertion/3
      - binary_input_assertion/2
      - clean_binary_input/0
      - set_text_output/3
      - set_text_output/2
      - set_text_output/1
      - check_text_output/3
      - check_text_output/2
      - check_text_output/1
      - text_output_assertion/4
      - text_output_assertion/3
      - text_output_assertion/2
      - text_output_contents/3
      - text_output_contents/2
      - text_output_contents/1
      - clean_text_output/0
      - set_binary_output/3
      - set_binary_output/2
      - set_binary_output/1
      - check_binary_output/2
      - check_binary_output/1
      - binary_output_assertion/3
      - binary_output_assertion/2
      - binary_output_contents/2
```

- *binary_output_contents/1*
- *clean_binary_output/0*
- *create_text_file/3*
- *create_text_file/2*
- *create_binary_file/2*
- *check_text_file/3*
- *check_text_file/2*
- *text_file_assertion/4*
- *text_file_assertion/3*
- *check_binary_file/2*
- *binary_file_assertion/3*
- *clean_file/1*
- *clean_directory/1*
- *closed_input_stream/2*
- *closed_output_stream/2*
- *stream_position/1*
- *test/2*

- *Private predicates*
  - *running_test_sets_/0*
  - *test/3*
  - *test_time_/3*
  - *auxiliary_predicate_counter_/1*
  - *test_/2*
  - *selected_test_/1*
  - *linter_warning_sequence_/1*
  - *recorded_linter_warning_/7*
  - *skipped_/1*
  - *passed_/3*
  - *failed_/3*
  - *flaky_/1*
  - *fired_/3*
  - *covered_/4*
- *Operators*
  - *op(700,xfx,=~=)*

Public predicates

cover/1

Declares entities being tested for which code coverage information should be collected.

Compilation flags:
`static`

Template:
`cover(Entity)`
Mode and number of proofs:
`cover(?entity_identifier)` - `zero_or_more`

run/0

Runs the unit tests, writing the results to the current output stream.

Compilation flags:
`static`

Mode and number of proofs:
`run` - `one`

---

run/1

Runs a unit test or a list of unit tests, writing the results to the current output stream. Runs the global setup and cleanup steps when defined. Fails when given a partial list of tests or when one of the test identifiers is not valid.

Compilation flags:
`static`

Template:
`run(Tests)`
Mode and number of proofs:
`run(++callable)` - `zero_or_one`
`run(++list(callable))` - `zero_or_one`

---

run/2

Runs the unit tests, writing the results to the specified file. Mode can be either `write` (to create a new file) or `append` (to add results to an existing file).

Compilation flags:
`static`

Template:
`run(File,Mode)`
Mode and number of proofs:
`run(+atom,+atom)` - `one`

---

run_test_sets/1

Runs two or more test sets as a unified set generating a single code coverage report if one is requested. When there is a single test set, it is equivalent to sending the message `run/0` to the test set. Trivially succeeds when the argument is an empty list.

Compilation flags:
`static`

Template:
`run_test_sets(TestObjects)`
Mode and number of proofs:
`run_test_sets(+list(object))` - `one`

Exceptions:
`TestObjects` is a partial list or a list with an element which is a variable:
`instantiation_error`
`TestObjects` is neither a partial list nor a list:
`type_error(list(object),TestObjects)`
An element `TestObject` of the `TestObjects` list is not an existing object:
`existence_error(object,TestObject)`

---

test/1

Enumerates, by backtracking, the identifiers of all defined unit tests.

Compilation flags:
`static`

Template:
`test(Identifier)`
Mode and number of proofs:
`test(?callable)` - `zero_or_more`

---

number_of_tests/1

Number of defined unit tests.

Compilation flags:
`static`

Template:
`number_of_tests(NumerOfTests)`
Mode and number of proofs:
`number_of_tests(?integer)` - `zero_or_one`

---

deterministic/1

True if the goal succeeds once without leaving choice-points.

Compilation flags:
`static`

Template:
`deterministic(Goal)`
Meta-predicate template:
`deterministic(0)`
Mode and number of proofs:
`deterministic(+callable)` - `zero_or_one`

---

deterministic/2

Reified version of the `deterministic/1` predicate. True if the goal succeeds. Returns a boolean value (`true` or `false`) indicating if the goal succeeded without leaving choice-points.

Compilation flags:
`static`

Template:
`deterministic(Goal,Deterministic)`
Meta-predicate template:
`deterministic(0,*)`
Mode and number of proofs:
`deterministic(+callable,--atom)` - `zero_or_one`

---

assertion/1

True if the assertion goal succeeds. Throws an error using the assertion goal as argument if the assertion goal throws an error or fails.

Compilation flags:
`static`

Template:
`assertion(Assertion)`
Meta-predicate template:
`assertion(::)`
Mode and number of proofs:
`assertion(@callable)` - `one`

Exceptions:

Assertion goal fails:

```
assertion_failure(Assertion)
```

Assertion goal throws Error:

```
assertion_error(Assertion,Error)
```

---

assertion/2

True if the assertion goal succeeds. Throws an error using the description as argument if the assertion goal throws an error or fails. The description argument helps to distinguish between different assertions in the same test body.

Compilation flags:

```
static
```

Template:

```
assertion(Description,Assertion)
```

Meta-predicate template:

```
assertion(*,0)
```

Mode and number of proofs:

```
assertion(+nonvar,@callable) - one
```

Exceptions:

Assertion goal fails:

```
assertion_failure(Description)
```

Assertion goal throws Error:

```
assertion_error(Description,Error)
```

---

quick_check/3

Reified version of the `quick_check/2` predicate. Reports `passed(SequenceSeed,Discarded,Labels)`, `failed(Goal,SequenceSeed,TestSeed)`, `error(Error,Goal,SequenceSeed,TestSeed)`, or `broken(Why,Culprit)`. `Goal` is the failed test.

Compilation flags:

```
static
```

Template:

```
quick_check(Template,Result,Options)
```

Meta-predicate template:

```
quick_check(::,*,::)
```

Mode and number of proofs:

```
quick_check(@callable,-callable,++list(compound)) - one
```

Remarks:

- `SequenceSeed` argument: Can be used to re-run the same exact sequence of pseudo-random tests by using the `rs/1` option after changes to the code being tested.

- `TestSeed` argument: Can be used to re-run the test that failed by using the `rs/1` option after changes to the code being tested.
- `Discarded` argument: Number of generated tests that were discarded for failing to comply a pre-condition specified using the `pc/1` option.
- `Labels` argument: List of pairs `Label-N` where `N` is the number of generated tests that are classified as `Label` by a closure specified using the `l/1` option.
- `broken(Why,Culprit)` result: This result signals a broken setup. For example, an invalid template, a broken pre-condition or label goal, or broken test generation.

---

quick_check/2

Generates and runs random tests for a predicate given its mode template and a set of options. Fails when a generated test fails printing the test. Also fails on an invalid option, printing the option.

Compilation flags:

```
static
```

Template:

```
quick_check(Template,Options)
```

Meta-predicate template:

```
quick_check(::,::)
```

Mode and number of proofs:

```
quick_check(@callable,++list(compound)) - zero_or_one
```

Remarks:

- Number of tests: Use the `n(NumberOfTests)` option to specify the number of random tests. Default is 100.
- Maximum number of shrink operations: Use the `s(MaxShrinks)` option to specify the number of shrink operations when a counter example is found. Default is 64.
- Type edge cases: Use the `ec(Boolean)` option to specify if type edge cases are tested (before generating random tests). Default is `true`.
- Starting seed: Use the `rs(Seed)` option to specify the random generator starting seed to be used when generating tests. No default. Seeds should be regarded as opaque terms.
- Test generation filtering: Use the `pc/1` option to specify a pre-condition closure for filtering generated tests (extended with the test arguments; no default).
- Generated tests classification: Use the `l/1` option to specify a label closure for classifying the generated tests (extended with the test arguments plus the labels argument; no default). The labelling predicate can return a single test label or a list of test labels.
- Verbose test generation: Use the `v(Boolean)` option to specify verbose reporting of generated random tests. Default is `false`.
- Progress bar: Use the `pb(Boolean,Tick)` option to print a progress bar for the executed tests, advancing at every `Tick` tests. Default is `false`. Only applies when the verbose option is false.

---

quick_check/1

Generates and runs random tests using default options for a predicate given its mode template. Fails when a generated test fails printing the test.

Compilation flags:
`static`

Template:
`quick_check(Template)`
Mode and number of proofs:
`quick_check(@callable)` - `zero_or_one`

---

benchmark/2

Benchmarks a goal and returns the total execution time in seconds. Uses CPU clock. Goals that may throw an exception should be wrapped by the `catch/3` control construct.

Compilation flags:
`static`

Template:
`benchmark(Goal,Time)`
Meta-predicate template:
`benchmark(0,*)`
Mode and number of proofs:
`benchmark(+callable,-float)` - `one`

---

benchmark_reified/3

Benchmarks a goal and returns the total execution time in seconds plus its result (`success`, `failure`, or `error(Error)`). Uses CPU clock.

Compilation flags:
`static`

Template:
`benchmark_reified(Goal,Time,Result)`
Meta-predicate template:
`benchmark_reified(0,*,*)`
Mode and number of proofs:
`benchmark_reified(+callable,-float,-callable)` - `one`

---

benchmark/3

Benchmarks a goal by repeating it the specified number of times and returning the total execution time in seconds. Uses CPU clock. Goals that may throw an exception should be wrapped by the `catch/3` control construct.

Compilation flags:
`static`

Template:
`benchmark(Goal,Repetitions,Time)`
Meta-predicate template:
`benchmark(0,*,*)`
Mode and number of proofs:
`benchmark(@callable,+positive_integer,-float) - one`

---

benchmark/4

Benchmarks a goal by repeating it the specified number of times and returning the total execution time in seconds using the given clock (`cpu` or `wall`). Goals that may throw an exception should be wrapped by the `catch/3` control construct.

Compilation flags:
`static`

Template:
`benchmark(Goal,Repetitions,Clock,Time)`
Meta-predicate template:
`benchmark(0,*,*,*)`
Mode and number of proofs:
`benchmark(@callable,+positive_integer,+atom,-float) - one`

---

variant/2

True when the two arguments are a variant of each other. I.e. if is possible to rename the term variables to make them identical. Useful for checking expected test results that contain variables.

Compilation flags:
`static`

Template:
`variant(Term1,Term2)`
Mode and number of proofs:
`variant(@term,@term) - zero_or_one`

---

approximately_equal/2

Compares two numbers for approximate equality given the `epsilon` arithmetic constant value using the de facto standard formula `abs(Number1 - Number2) =< max(abs(Number1), abs(Number2)) * epsilon`. Type-checked.

Compilation flags:

`static`

Template:

`approximately_equal(Number1,Number2)`

Mode and number of proofs:

`approximately_equal(+number,+number)` - `zero_or_one`

---

approximately_equal/3

Compares two numbers for approximate equality given a user-defined epsilon value using the de facto standard formula `abs(Number1 - Number2) =< max(abs(Number1), abs(Number2)) * Epsilon`. Type-checked.

Compilation flags:

`static`

Template:

`approximately_equal(Number1,Number2,Epsilon)`

Mode and number of proofs:

`approximately_equal(+number,+number,+number)` - `zero_or_one`

Remarks:

- Epsilon range: Epsilon should be the `epsilon` arithmetic constant value or a small multiple of it. Only use a larger value if a greater error is expected.
- Comparison with essential equality: For the same epsilon value, approximate equality is weaker requirement than essential equality.

---

essentially_equal/3

Compares two numbers for essential equality given an epsilon value using the de facto standard formula `abs(Number1 - Number2) =< min(abs(Number1), abs(Number2)) * Epsilon`. Type-checked.

Compilation flags:

`static`

Template:

`essentially_equal(Number1,Number2,Epsilon)`

Mode and number of proofs:

`essentially_equal(+number,+number,+number)` - `zero_or_one`

Remarks:

- Comparison with approximate equality: For the same epsilon value, essential equality is a stronger requirement than approximate equality.

---

tolerance_equal/4

Compares two numbers for close equality given relative and absolute tolerances using the de facto standard formula `abs(Number1 - Number2) =< max(RelativeTolerance * max(abs(Number1), abs(Number2)), AbsoluteTolerance)`. Type-checked.

Compilation flags:

`static`

Template:

`tolerance_equal(Number1,Number2,RelativeTolerance,AbsoluteTolerance)`

Mode and number of proofs:

`tolerance_equal(+number,+number,+number,+number)` - `zero_or_one`

=~= / 2

Compares two numbers (or lists of numbers) for approximate equality using `100*epsilon` for the absolute error and, if that fails, `99.999%` accuracy for the relative error. But these precision values may not be adequate for all cases. Type-checked.

Compilation flags:

`static`

Template:

`=~=(Number1,Number2)`

Mode and number of proofs:

`=~=(+number,+number)` - `zero_or_one`

`=~=(+list(number),+list(number))` - `zero_or_one`

---

epsilon/1

Returns the value of epsilon used in the definition of the `(=~=)/2` predicate.

Compilation flags:

`static`

Template:

`epsilon(Epsilon)`

Mode and number of proofs:

`epsilon(-float)` - `one`

Protected predicates

run_tests/0

Runs all defined unit tests.

Compilation flags:
`static`

Mode and number of proofs:
`run_tests` - `one`

---

run_tests/1

Runs all the tests defined in the given file.

Compilation flags:
`static`

Template:
`run_tests(File)`
Mode and number of proofs:
`run_tests(+atom)` - `one`

---

run_test_set/0

Runs a test set as part of running two or more test sets as a unified set.

Compilation flags:
`static`

Mode and number of proofs:
`run_test_set` - `one`

---

run_quick_check_tests/5

Runs a QuickCheck test using the given options. Returns the starting seed used to generate the random tests, the number of discarded tests, and the test label statistics.

Compilation flags:
`static`

Template:

`run_quick_check_tests(Template,Options,Seed,Discarded,Labels)`

Meta-predicate template:

`run_quick_check_tests(::,::,*,*,*)`

Mode and number of proofs:

`run_quick_check_tests(@callable,+list,--nonvar,--number,--list(pair))` - `one_or_error`

Exceptions:

`Template` is a variable:

`quick_check_broken(template_error(Template),instantiation_error)`

`Template` is not a callable term:

`quick_check_broken(template_error(Template),type_error(callable,Template))`

An error is raised while checking a generated test pre-condition:

`quick_check_broken(pre_condition_error(Condition),Error)`

All generated tests fail the pre-condition:

`quick_check_broken(pre_condition_always_fails(Condition))`

An error is raised while evaluating a label goal:

`quick_check_broken(label_goal_error(Label),Error)`

A label goal fails:

`quick_check_broken(label_goal_failure(Label))`

An error is raised while generating test arguments:

`quick_check_broken(generate_test_error(Template),Error)`

Test argument generation fails:

`quick_check_broken(generate_test_failure(Template),Type)`

---

condition/0

Verifies conditions for running the tests. Defaults to the goal `true`.

Compilation flags:

`static`

Mode and number of proofs:

`condition` - `zero_or_one`

---

setup/0

Setup environment before running the test set. Defaults to the goal `true`.

Compilation flags:

`static`

Mode and number of proofs:

`setup` - `zero_or_one`

---

cleanup/0

Cleanup environment after running the test set. Defaults to the goal `true`.

Compilation flags:
`static`

Mode and number of proofs:
`cleanup` - `zero_or_one`

---

make/1

Make target for automatically running the test set when calling the `logtalk_make/1` built-in predicate. No default. Possible values are `all` and `check`.

Compilation flags:
`static`

Template:
`make(Target)`
Mode and number of proofs:
`make(?atom)` - `zero_or_one`

---

note/1

Note to be printed after the test results. Defaults to the empty atom.

Compilation flags:
`static`

Template:
`note(Note)`
Mode and number of proofs:
`note(?atom)` - `zero_or_one`

---

file_path/2

Returns the absolute path for a file path that is relative to the tests object path. When the file path is already an absolute path, it is expanded to resolve any remaining relative file path parts.

Compilation flags:
`static`

Template:

```
file_path(File,Path)
```

Mode and number of proofs:

```
file_path(+atom,-atom) - one
```

See also:

clean_file/1
clean_directory/1

---

file_url/2

Returns the URL for a file path that can be either absolute or relative to the tests object path. When the file path is an absolute path, it is expanded to resolve any remaining relative file path parts.

Compilation flags:

```
static
```

Template:

```
file_url(File,URL)
```

Mode and number of proofs:

```
file_url(+atom,-atom) - one
```

---

suppress_text_output/0

Suppresses text output. Useful to avoid irrelevant text output from predicates being tested to clutter the test logs.

Compilation flags:

```
static
```

Mode and number of proofs:

```
suppress_text_output - one
```

---

suppress_binary_output/0

Suppresses binary output. Useful to avoid irrelevant binary output from predicates being tested to clutter the test logs.

Compilation flags:

```
static
```

Mode and number of proofs:

```
suppress_binary_output - one
```

---

set_text_input/3

Creates a temporary file, in the same directory as the tests object, with the given text contents, and opens it for reading referenced by the given alias and using the additional options. If no `eof_action/1` option is specified, its value will be the default used by the backend compiler.

Compilation flags:

`static`

Template:

`set_text_input(Alias,Contents,Options)`

Mode and number of proofs:

`set_text_input(+atom,+atom,+list(stream_option))` - `one`

`set_text_input(+atom,+list(atom),+list(stream_option))` - `one`

See also:

text_input_assertion/3

check_text_input/2

clean_text_input/0

---

set_text_input/2

Creates a temporary file, in the same directory as the tests object, with the given text contents, and opens it for reading referenced by the given alias and using the default end-of-file action for the used backend compiler.

Compilation flags:

`static`

Template:

`set_text_input(Alias,Contents)`

Mode and number of proofs:

`set_text_input(+atom,+atom)` - `one`

`set_text_input(+atom,+list(atom))` - `one`

See also:

text_input_assertion/3

check_text_input/2

clean_text_input/0

---

set_text_input/1

Creates a temporary file, in the same directory as the tests object, with the given text contents, opens it for reading using the default end-of-file action for the used backend compiler, and sets the current input stream to the file.

Compilation flags:

```
static
```

Template:

```
set_text_input(Contents)
```

Mode and number of proofs:

```
set_text_input(+atom) - one
set_text_input(+list(atom)) - one
```

See also:

text_input_assertion/2
check_text_input/1
clean_text_input/0

---

check_text_input/2

Checks that the temporary file (referenced by the given alias) being read have the expected text contents.

Compilation flags:

```
static
```

Template:

```
check_text_input(Alias,Contents)
```

Mode and number of proofs:

```
check_text_input(+atom,+atom) - zero_or_one
```

See also:

set_text_input/2
set_text_input/2
text_input_assertion/3
clean_text_input/0

---

check_text_input/1

Checks that the temporary file being read have the expected text contents.

Compilation flags:

```
static
```

Template:

```
check_text_input(Contents)
```

Mode and number of proofs:

```
check_text_input(+atom) - zero_or_one
```

See also:

set_text_input/1
text_input_assertion/2
clean_text_input/0

---

text_input_assertion/3

Returns an assertion for checking that the temporary file (referenced by the given alias) being read have the expected text contents.

Compilation flags:

```
static
```

Template:

```
text_input_assertion(Alias,Contents,Assertion)
```

Mode and number of proofs:

```
text_input_assertion(+atom,+atom,--callable) - one
```

See also:

set_text_input/3
check_text_input/2
clean_text_input/0

---

text_input_assertion/2

Returns an assertion for checking that the temporary file being read have the expected text contents.

Compilation flags:

```
static
```

Template:

```
text_input_assertion(Contents,Assertion)
```

Mode and number of proofs:

```
text_input_assertion(+atom,--callable) - one
```

See also:

set_text_input/1
check_text_input/1
clean_text_input/0

---

clean_text_input/0

Cleans the temporary file used when testing text input.

Compilation flags:
`static`

Mode and number of proofs:
`clean_text_input` - `one`

See also:
set_text_input/3
set_text_input/2
set_text_input/1

---

set_binary_input/3

Creates a temporary file, in the same directory as the tests object, with the given binary contents, and opens it for reading referenced by the given alias and using the additional options. If no `eof_action/1` option is specified, its value will be the default used by the backend compiler.

Compilation flags:
`static`

Template:
`set_binary_input(Alias,Bytes,Options)`
Mode and number of proofs:
`set_binary_input(+atom,+list(byte),+list(stream_option))` - `one`

See also:
binary_input_assertion/3
check_binary_input/2
clean_binary_input/0

---

set_binary_input/2

Creates a temporary file, in the same directory as the tests object, with the given binary contents, and opens it for reading referenced by the given alias and using the default end-of-file action for the used backend compiler.

Compilation flags:
`static`

Template:
`set_binary_input(Alias,Bytes)`
Mode and number of proofs:

`set_binary_input(+atom,+list(byte))` - `one`

See also:

binary_input_assertion/3
check_binary_input/2
clean_binary_input/0

---

set_binary_input/1

Creates a temporary file, in the same directory as the tests object, with the given binary contents, and opens it for reading using the default end-of-file action for the used backend compiler, and sets the current input stream to the file.

Compilation flags:

`static`

Template:

`set_binary_input(Bytes)`

Mode and number of proofs:

`set_binary_input(+list(byte))` - `one`

See also:

binary_input_assertion/2
check_binary_input/1
clean_binary_input/0

---

check_binary_input/2

Checks that the temporary file (referenced by the given alias) being read have the expected binary contents.

Compilation flags:

`static`

Template:

`check_binary_input(Alias,Bytes)`

Mode and number of proofs:

`check_binary_input(+atom,+list(byte))` - `zero_or_one`

See also:

set_binary_input/3
set_binary_input/2
binary_input_assertion/3
clean_binary_input/0

---

check_binary_input/1

Checks that the temporary file being read have the expected binary contents.

Compilation flags:
`static`

Template:
`check_binary_input(Bytes)`
Mode and number of proofs:
`check_binary_input(+list(byte))` - `zero_or_one`

See also:
binary_input_assertion/2
set_binary_input/1
clean_binary_input/0

---

binary_input_assertion/3

Returns an assertion for checking that the temporary file (referenced by the given alias) being read have the expected binary contents.

Compilation flags:
`static`

Template:
`binary_input_assertion(Alias,Bytes,Assertion)`
Mode and number of proofs:
`binary_input_assertion(+atom,+list(byte),--callable)` - `one`

See also:
check_binary_input/2
set_binary_input/3
set_binary_input/2
clean_binary_input/0

---

binary_input_assertion/2

Returns an assertion for checking that the temporary file being read have the expected binary contents.

Compilation flags:
`static`

Template:
`binary_input_assertion(Bytes,Assertion)`
Mode and number of proofs:

`binary_input_assertion(+list(byte),--callable)` - `one`

See also:

check_binary_input/1
set_binary_input/1
clean_binary_input/0

---

clean_binary_input/0

Cleans the temporary file used when testing binary input.

Compilation flags:

`static`

Mode and number of proofs:

`clean_binary_input` - `one`

See also:

set_binary_input/3
set_binary_input/2
set_binary_input/1

---

set_text_output/3

Creates a temporary file, in the same directory as the tests object, with the given text contents, and opens it for writing referenced by the given alias and using the additional options.

Compilation flags:

`static`

Template:

`set_text_output(Alias,Contents,Options)`

Mode and number of proofs:

`set_text_output(+atom,+atom,+list(stream_option))` - `one`
`set_text_output(+atom,+list(atom),+list(stream_option))` - `one`

See also:

text_output_assertion/4
check_text_output/3
clean_text_output/0

---

set_text_output/2

Creates a temporary file, in the same directory as the tests object, with the given text contents, and referenced by the given alias.

Compilation flags:

```
static
```

Template:

```
set_text_output(Alias,Contents)
```

Mode and number of proofs:

```
set_text_output(+atom,+atom) - one
set_text_output(+atom,+list(atom)) - one
```

See also:

text_output_assertion/3
check_text_output/2
clean_text_output/0

---

set_text_output/1

Creates a temporary file, in the same directory as the tests object, with the given text contents, and sets the current output stream to the file.

Compilation flags:

```
static
```

Template:

```
set_text_output(Contents)
```

Mode and number of proofs:

```
set_text_output(+atom) - one
set_text_output(+list(atom)) - one
```

See also:

text_output_assertion/2
check_text_output/1
clean_text_output/0

---

check_text_output/3

Checks that the temporary file (open with the given options and alias in the same directory as the tests object) being written have the expected text contents.

Compilation flags:

```
static
```

Template:
`check_text_output(Alias,Contents,Options)`
Mode and number of proofs:
`check_text_output(+atom,+atom,+list(stream_option))` - `zero_or_one`

See also:
set_text_output/3
text_output_assertion/4
clean_text_output/0

---

check_text_output/2

Checks that the temporary file (open with default options and alias in the same directory as the tests object) being written have the expected text contents.

Compilation flags:
`static`

Template:
`check_text_output(Alias,Contents)`
Mode and number of proofs:
`check_text_output(+atom,+atom)` - `zero_or_one`

See also:
set_text_output/2
text_output_assertion/3
clean_text_output/0

---

check_text_output/1

Checks that the temporary file being written have the expected text contents.

Compilation flags:
`static`

Template:
`check_text_output(Contents)`
Mode and number of proofs:
`check_text_output(+atom)` - `zero_or_one`

See also:
set_text_output/1
text_output_assertion/2
clean_text_output/0

---

text_output_assertion/4

Returns an assertion for checking that the temporary file (open with the given options and alias in the same directory as the tests object) being written have the expected text contents.

Compilation flags:

```
static
```

Template:

```
text_output_assertion(Alias,Contents,Options,Assertion)
```

Mode and number of proofs:

```
text_output_assertion(+atom,+atom,+list(stream_option),--callable) - one
```

See also:

set_text_output/3
check_text_output/3
clean_text_output/0

---

text_output_assertion/3

Returns an assertion for checking that the temporary file (open with default options and alias in the same directory as the tests object) being written have the expected text contents.

Compilation flags:

```
static
```

Template:

```
text_output_assertion(Alias,Contents,Assertion)
```

Mode and number of proofs:

```
text_output_assertion(+atom,+atom,--callable) - one
```

See also:

set_text_output/2
check_text_output/2
clean_text_output/0

---

text_output_assertion/2

Returns an assertion for checking that the temporary file (open with default options in the same directory as the tests object) being written have the expected text contents.

Compilation flags:

```
static
```

Template:

```
text_output_assertion(Contents,Assertion)
```

Mode and number of proofs:

```
text_output_assertion(+atom,--callable) - one
```

See also:

set_text_output/1
check_text_output/1
clean_text_output/0

---

text_output_contents/3

Returns the contents of the temporary file (open with the given options and alias in the same directory as the tests object) being written.

Compilation flags:

```
static
```

Template:

```
text_output_contents(Alias,Contents,Options)
```

Mode and number of proofs:

```
text_output_contents(+atom,-list(character),+list(stream_option)) - one
```

---

text_output_contents/2

Returns the contents of the temporary file (open with default options and alias in the same directory as the tests object) being written.

Compilation flags:

```
static
```

Template:

```
text_output_contents(Alias,Contents)
```

Mode and number of proofs:

```
text_output_contents(+atom,-list(character)) - one
```

---

text_output_contents/1

Returns the contents of the temporary file (open with default options in the same directory as the tests object) being written.

Compilation flags:

```
static
```

Template:

```
text_output_contents(Contents)
```

Mode and number of proofs:

```
text_output_contents(-list(character)) - one
```

---

clean_text_output/0

Cleans the temporary file used when testing text output.

Compilation flags:

```
static
```

Mode and number of proofs:

```
clean_text_output - one
```

See also:

set_text_output/3
set_text_output/2
set_text_output/1

---

set_binary_output/3

Creates a temporary file, in the same directory as the tests object, with the given binary contents, and opens it for writing referenced by the given alias and using the additional options.

Compilation flags:

```
static
```

Template:

```
set_binary_output(Alias,Contents,Options)
```

Mode and number of proofs:

```
set_binary_output(+atom,+list(byte),+list(stream_option)) - one
```

See also:

binary_output_assertion/3
check_binary_output/2
clean_binary_output/0

---

set_binary_output/2

Creates a temporary file, in the same directory as the tests object, with the given binary contents, and opens it for writing referenced with the given alias.

Compilation flags:

```
static
```

Template:

```
set_binary_output(Alias,Bytes)
```

Mode and number of proofs:

```
set_binary_output(+atom,+list(byte)) - one
```

See also:

binary_output_assertion/3
check_binary_output/2
clean_binary_output/0

---

set_binary_output/1

Creates a temporary file, in the same directory as the tests object, with the given binary contents, and sets the current output stream to the file.

Compilation flags:

```
static
```

Template:

```
set_binary_output(Bytes)
```

Mode and number of proofs:

```
set_binary_output(+list(byte)) - one
```

See also:

binary_output_assertion/2
check_binary_output/1
clean_binary_output/0

---

check_binary_output/2

Checks that the temporary file (referenced by the given alias) have the expected binary contents.

Compilation flags:

```
static
```

Template:

```
check_binary_output(Alias,Bytes)
```

Mode and number of proofs:

`check_binary_output(+atom,+list(byte))` - `zero_or_one`

See also:

set_binary_output/3
set_binary_output/2
binary_output_assertion/3
clean_binary_output/0

---

check_binary_output/1

Checks that the temporary file (open in the same directory as the tests object) have the expected binary contents.

Compilation flags:

`static`

Template:

`check_binary_output(Bytes)`

Mode and number of proofs:

`check_binary_output(+list(byte))` - `zero_or_one`

See also:

set_binary_output/1
binary_output_assertion/2
clean_binary_output/0

---

binary_output_assertion/3

Returns an assertion for checking that the temporary file (referenced by the given alias) have the expected binary contents.

Compilation flags:

`static`

Template:

`binary_output_assertion(Alias,Bytes,Assertion)`

Mode and number of proofs:

`binary_output_assertion(+atom,+list(byte),--callable)` - `one`

See also:

set_binary_output/2
check_binary_output/2
clean_binary_output/0

---

binary_output_assertion/2

Returns an assertion for checking that the temporary file (open in the same directory as the tests object) have the expected binary contents.

Compilation flags:
`static`

Template:
`binary_output_assertion(Bytes,Assertion)`
Mode and number of proofs:
`binary_output_assertion(+list(byte),--callable)` - `one`

See also:
set_binary_output/1
check_binary_output/1
clean_binary_output/0

---

binary_output_contents/2

Returns the binary contents of the temporary file (referenced by the given alias) being written.

Compilation flags:
`static`

Template:
`binary_output_contents(Alias,Bytes)`
Mode and number of proofs:
`binary_output_contents(+atom,-list(byte))` - `one`

---

binary_output_contents/1

Returns the binary contents of the temporary file being written.

Compilation flags:
`static`

Template:
`binary_output_contents(Bytes)`
Mode and number of proofs:
`binary_output_contents(-list(byte))` - `one`

---

clean_binary_output/0

Cleans the temporary file used when testing binary output.

Compilation flags:

```
static
```

Mode and number of proofs:

```
clean_binary_output - one
```

See also:

set_binary_output/3
set_binary_output/2
set_binary_output/1

---

create_text_file/3

Creates a text file with the given contents. The file is open for writing using the given options. Relative file paths are interpreted as relative to the tests object path.

Compilation flags:

```
static
```

Template:

```
create_text_file(File,Contents,Options)
```

Mode and number of proofs:

```
create_text_file(+atom,+atom,+list(stream_option)) - one
create_text_file(+atom,+list(atom),+list(stream_option)) - one
```

---

create_text_file/2

Creates a text file with the given contents. The file is open for writing using default options. Relative file paths are interpreted as relative to the tests object path.

Compilation flags:

```
static
```

Template:

```
create_text_file(File,Contents)
```

Mode and number of proofs:

```
create_text_file(+atom,+atom) - one
create_text_file(+atom,+list(atom)) - one
```

---

create_binary_file/2

Creates a binary file with the given contents. Relative file paths are interpreted as relative to the tests object path.

Compilation flags:
`static`

Template:
`create_binary_file(File,Bytes)`
Mode and number of proofs:
`create_binary_file(+atom,+list(byte))` - `one`

---

check_text_file/3

Checks that the contents of a text file match the expected contents. The file is open for reading using the given options. Relative file paths are interpreted as relative to the tests object path.

Compilation flags:
`static`

Template:
`check_text_file(File,Contents,Options)`
Mode and number of proofs:
`check_text_file(+atom,+atom,+list(stream_option))` - `zero_or_one`

See also:
text_file_assertion/4

---

check_text_file/2

Checks that the contents of a text file (open for reading using default options) match the expected contents. Relative file paths are interpreted as relative to the tests object path.

Compilation flags:
`static`

Template:
`check_text_file(File,Contents)`
Mode and number of proofs:
`check_text_file(+atom,+atom)` - `zero_or_one`

See also:
text_file_assertion/3

---

text_file_assertion/4

Returns an assertion for checking that the given file have the expected text contents. The file is open for reading using the given options. Relative file paths are interpreted as relative to the tests object path.

Compilation flags:

```
static
```

Template:

```
text_file_assertion(File,Contents,Options,Assertion)
```

Mode and number of proofs:

```
text_file_assertion(+atom,+atom,+list(stream_option),--callable) - one
```

See also:

check_text_file/3

---

text_file_assertion/3

Returns an assertion for checking that the given file have the expected text contents. The file is open for reading using default options. Relative file paths are interpreted as relative to the tests object path.

Compilation flags:

```
static
```

Template:

```
text_file_assertion(File,Contents,Assertion)
```

Mode and number of proofs:

```
text_file_assertion(+atom,+atom,--callable) - one
```

See also:

check_text_file/2

---

check_binary_file/2

Checks the contents of a binary file match the expected contents. Relative file paths are interpreted as relative to the tests object path.

Compilation flags:

```
static
```

Template:

```
check_binary_file(File,Bytes)
```

Mode and number of proofs:

```
check_binary_file(+atom,+list(byte)) - zero_or_one
```

See also:

binary_file_assertion/3

---

binary_file_assertion/3

Returns an assertion for checking that the given file have the expected binary contents. Relative file paths are interpreted as relative to the tests object path.

Compilation flags:
`static`

Template:
`binary_file_assertion(File,Bytes,Assertion)`
Mode and number of proofs:
`binary_file_assertion(+atom,+list(byte),--callable)` - `one`

See also:
check_binary_file/2

---

clean_file/1

Closes any existing stream associated with the file and deletes the file if it exists. Relative file paths are interpreted as relative to the tests object path.

Compilation flags:
`static`

Template:
`clean_file(File)`
Mode and number of proofs:
`clean_file(+atom)` - `one`

See also:
clean_directory/1
file_path/2

---

clean_directory/1

Deletes a directory if it exists. Relative directory paths are interpreted as relative to the tests object path.

Compilation flags:
`static`

Template:

```
clean_directory(Directory)
```

Mode and number of proofs:

```
clean_directory(+atom) - one
```

See also:

clean_file/1
file_path/2

---

closed_input_stream/2

Opens a temporary file in the same directory as the tests object with the given options for reading, closes it, and returns its stream handle.

Compilation flags:

```
static
```

Template:

```
closed_input_stream(Stream,Options)
```

Mode and number of proofs:

```
closed_input_stream(-stream,+list(stream_option)) - one
```

---

closed_output_stream/2

Opens a temporary file in the same directory as the tests object with the given options for writing, closes it, and returns its stream handle.

Compilation flags:

```
static
```

Template:

```
closed_output_stream(Stream,Options)
```

Mode and number of proofs:

```
closed_output_stream(-stream,+list(stream_option)) - zero_or_one
```

---

stream_position/1

Returns a syntactically valid stream position by opening a temporary file in the same directory as the tests object.

Compilation flags:

```
static
```

Template:

```
stream_position(Position)
```

Mode and number of proofs:

```
stream_position(-stream_position) - one
```

---

test/2

Table of defined tests.

Compilation flags:

```
static
```

Template:

```
test(Identifier,Test)
```

Mode and number of proofs:

```
test(?callable,?compound) - zero_or_more
```

Private predicates

running_test_sets_/0

Internal flag used when running two or more test sets as a unified set.

Compilation flags:

```
dynamic
```

Mode and number of proofs:

```
running_test_sets_ - zero_or_one
```

---

test/3

Compiled unit tests. The list of variables is used to ensure variable sharing between a test with its test options.

Compilation flags:

```
static
```

Template:

```
test(Identifier,Variables,Outcome)
```

Mode and number of proofs:

```
test(?callable,?list(variable),?nonvar) - zero_or_more
```

---

test_time_/3

Test identifiers, objects, and wall times for the current test run.

Compilation flags:
```
dynamic
```

Template:
```
test_time_(Test,Object,WallTime)
```
Mode and number of proofs:
```
test_time_(?callable,?object,?float) - zero_or_more
```

---

auxiliary_predicate_counter_/1

Counter for generating unique auxiliary predicate names.

Compilation flags:
```
dynamic
```

Template:
```
auxiliary_predicate_counter_(Counter)
```
Mode and number of proofs:
```
auxiliary_predicate_counter_(?integer) - one_or_more
```

---

test_/2

Table of compiled tests.

Compilation flags:
```
dynamic
```

Template:
```
test_(Identifier,Test)
```
Mode and number of proofs:
```
test_(?callable,?compound) - zero_or_more
```

---

selected_test_/1

Table of selected tests for execution.

Compilation flags:
dynamic

Template:
selected_test_(Identifier)
Mode and number of proofs:
selected_test_(?callable) - zero_or_more

---

linter_warning_sequence_/1

Counter for cached test-compilation linter warnings.

Compilation flags:
dynamic

Template:
linter_warning_sequence_(Sequence)
Mode and number of proofs:
linter_warning_sequence_(?integer) - zero_or_one

---

recorded_linter_warning_/7

Cache of test-compilation linter warnings.

Compilation flags:
dynamic

Template:
recorded_linter_warning_(Sequence,RuleId,Message,Context,File,Lines,Properties)
Mode and number of proofs:
recorded_linter_warning_(?integer,?atom,?atom,?compound,?atom,?compound,?list(compound)) - zero_or_more

---

skipped_/1

Counter for skipped tests.

Compilation flags:
`dynamic`

Template:
`skipped_(Counter)`
Mode and number of proofs:
`skipped_(?integer)` - `zero_or_one`

---

passed_/3

Counter and total time for passed tests.

Compilation flags:
`dynamic`

Template:
`passed_(Counter,CPUTime,WallTime)`
Mode and number of proofs:
`passed_(?integer,-float,-float)` - `zero_or_one`

---

failed_/3

Counter and total time for failed tests.

Compilation flags:
`dynamic`

Template:
`failed_(Counter,CPUTime,WallTime)`
Mode and number of proofs:
`failed_(?integer,-float,-float)` - `zero_or_one`

---

flaky_/1

Counter for failed tests that are marked as flaky.

Compilation flags:
`dynamic`

Template:
`flaky_(Counter)`
Mode and number of proofs:
`flaky_(?integer)` - `zero_or_one`

---

fired_/3

Fired clauses when running the unit tests.

Compilation flags:
`dynamic`

Template:
`fired_(Entity,Predicate,Clause)`
Mode and number of proofs:
`fired_(?entity_identifier,?predicate_indicator,?integer)` - `zero_or_more`

---

covered_/4

Auxiliary predicate for collecting statistics on clause coverage.

Compilation flags:
`dynamic`

Template:
`covered_(Entity,Predicate,Covered,Total)`
Mode and number of proofs:
`covered_(?entity_identifier,?callable,?integer,?integer)` - `zero_or_more`

Operators

`op(700,xfx,=~=)`

Scope:
public

category

1.198.8 `lgtunit_messages`

Logtalk unit test framework default message translations.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 12:5:0
Date: 2026-11-17

Compilation flags:
static

Provides:
logtalk::message_prefix_stream/4
logtalk::message_tokens//2

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *failed_test_reason//1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`failed_test_reason//1`

Used to rewrite a term representing the reason why a term failed into a list of tokens.

Compilation flags:
static

Template:

```
failed_test_reason(Reason)
```

Mode and number of proofs:

```
failed_test_reason(@nonvar) - one_or_more
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.198.9 `minimal_output`

Intercepts unit test execution messages and outputs a minimal report.

Availability:

```
logtalk_load(lgtunit(loader))
```

Author: Paulo Moura
Version: 3:0:0
Date: 2021-05-27

Compilation flags:

```
static, context_switching_calls
```

Provides:

logtalk::message_hook/4

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(minimal_output))`.
- Limitations: Cannot be used when the test objects also intercept `lgtunit` messages.

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.198.10 `sonar_report`

Intercepts unit test execution messages and generates a `sonar.xml` file with code coverage results using the SonarQube and SonarCloud Generic Test Coverage XML format.

Availability:

`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-18

Compilation flags:

`static, context_switching_calls`

Provides:

logtalk::message_hook/4

Uses:

logtalk
user

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(sonar_report))`.
- Analysis parameter: Pass the generated report to the Sonar scanner using the `sonar.coverageReportPaths` analysis parameter.
- File paths: Sonar requires file paths to be either absolute or relative to the project root. Use the `suppress_path_prefix` flag to strip the prefix of the source file paths when necessary.
- Lines to cover: Each predicate is reported as a single line to cover, the line of its first clause. The line is reported as covered when at least one of the predicate clauses is covered. Predicates without line information and with lines outside the source file are not reported.
- Branches to cover: Predicates with two or more clauses also report the number of clauses as the number of branches to cover and the number of covered clauses as the number of covered branches.
- Shared lines and files: Entities defined in the same source file are reported under a single file element. Predicates that share the same line are reported as a single line to cover.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *predicate_coverage_/4*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

predicate_coverage_/4

Table of predicate coverage data.

Compilation flags:
`dynamic`

Template:
`predicate_coverage_(File,Line,Covered,Total)`
Mode and number of proofs:
`predicate_coverage_(?atom,?integer,?integer,?integer)` - `zero_or_more`

Operators

(none)
object

1.198.11 subunit_v1_output

Intercepts unit test execution messages and outputs a Subunit v1 text stream to the current output stream.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-02

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
os

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(subunit_v1_output))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.198.12 `subunit_v1_report`

Intercepts unit test execution messages and generates a `subunit_v1_report.txt` file using the Subunit v1 text streaming format in the same directory as the tests object file.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-02

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
logtalk
os

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(subunit_v1_report))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.198.13 `subunit_v2_output`

Intercepts unit test execution messages and outputs a Subunit v2 binary stream to the current output stream.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-02

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4

Uses:
list
os
term_io
user

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load([basic_types(loader), term_io(loader),lgtunit(subunit_v2_output)])`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.198.14 subunit_v2_report

Intercepts unit test execution messages and generates a `subunit_v2_report.bin` file using the Subunit v2 binary streaming format in the same directory as the tests object file.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-02

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
list
logtalk
os
term_io
user

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load([basic_types(loader), term_io(loader),lgtunit(subunit_v2_report)])`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.198.15 `tap_output`

Intercepts unit test execution messages and outputs a report using the TAP format to the current output stream.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 5:0:0
Date: 2025-04-07

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(tap_output))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *generating_/0*
  - *partial_/1*
  - *test_count_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

generating_/0

Flag to detect report in progress when processing two or more test sets as a unified set.

Compilation flags:
dynamic

Mode and number of proofs:
generating_ - zero_or_one

---

partial_/1

Cache of total of tests per test set.

Compilation flags:
dynamic

Template:
partial_(Count)
Mode and number of proofs:
partial_(?integer) - zero_or_more

---

test_count_/1

Test counter.

Compilation flags:
dynamic

Template:
test_count_(Count)
Mode and number of proofs:
test_count_(?integer) - zero_or_one

Operators

(none)
object

1.198.16 tap_report

Intercepts unit test execution messages and generates a `tap_report.txt` file using the TAP output format in the same directory as the tests object file.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 6:0:0
Date: 2025-04-07

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
logtalk

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(tap_report))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *partial_/1*
  - *test_count_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

partial_/1

Cache of total of tests per test set.

Compilation flags:
`dynamic`

Template:
`partial_(Count)`
Mode and number of proofs:
`partial_(?integer)` - `zero_or_more`

---

test_count_/1

Test counter.

Compilation flags:
`dynamic`

Template:
`test_count_(Count)`
Mode and number of proofs:
`test_count_(?integer)` - `zero_or_one`

Operators

(none)
object

1.198.17 xunit_net_v2_output

Intercepts unit test execution messages and outputs a report using the xUnit.net v2 XML format to the current output stream.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 5:0:0
Date: 2025-04-07

Compilation flags:

`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
user

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(xunit_net_v2_output))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *message_cache_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

message_cache_/1

Table of messages emitted by the lgtunit tool when running tests.

Compilation flags:
`dynamic`

Template:
`message_cache_(Message)`
Mode and number of proofs:
`message_cache_(?callable)` - `zero_or_more`

Operators

(none)
object

1.198.18 xunit_net_v2_report

Intercepts unit test execution messages and generates a `xunit_report.xml` file using the xUnit.net v2 XML format in the same directory as the tests object file.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 6:0:0
Date: 2025-04-07

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
logtalk
user

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(xunit_net_v2_report))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *message_cache_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

message_cache_/1

Table of messages emitted by the lgtunit tool when running tests.

Compilation flags:
`dynamic`

Template:
`message_cache_(Message)`
Mode and number of proofs:
`message_cache_(?callable)` - `zero_or_more`

Operators

(none)
object

1.198.19 xunit_output

Intercepts unit test execution messages and outputs a report using the xUnit XML format to the current output stream.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 5:0:1
Date: 2025-04-11

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
logtalk
user

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(xunit_output))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - `message_cache_/1`
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`message_cache_/1`

Table of messages emitted by the lgtunit tool when running tests.

Compilation flags:
`dynamic`

Template:
`message_cache_(Message)`
Mode and number of proofs:
`message_cache_(?callable)` - `zero_or_more`

Operators

(none)
object

1.198.20 `xunit_report`

Intercepts unit test execution messages and generates a `xunit_report.xml` file using the xUnit XML format in the same directory as the tests object file.

Availability:
`logtalk_load(lgtunit(loader))`

Author: Paulo Moura
Version: 6:0:0
Date: 2025-04-07

Compilation flags:

`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
logtalk
user

Remarks:

- Usage: Simply load this object before running your tests using the goal `logtalk_load(lgtunit(xunit_report))`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *message_cache_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

message_cache_/1

Table of messages emitted by the lgtunit tool when running tests.

Compilation flags:
`dynamic`

Template:
`message_cache_(Message)`
Mode and number of proofs:
`message_cache_(?callable)` - `zero_or_more`

Operators

(none)

1.199 library

protocol

1.199.1 `cloning`

Object cloning protocol.

Availability:
`logtalk_load(library(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2010-09-14

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *clone/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`clone/1`

Clones an object, returning the identifier of the new object if none is given.

Compilation flags:
`static`

Template:
`clone(Clone)`

Mode and number of proofs:
`clone(?object)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.199.2 `counters`

Named integer counters. Counter names can be any nonvar term.

Availability:
`logtalk_load(library(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2022-02-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *counter/2*
  - *increment_counter/1*
  - *decrement_counter/1*
  - *reset_counter/1*
  - *reset_counters/0*
- *Protected predicates*

- *Private predicates*
  - *counter_/2*
- *Operators*

Public predicates

counter/2

True if `Counter` is a counter with value `Value`.

Compilation flags:
`static`

Template:
`counter(Counter,Value)`
Mode and number of proofs:
`counter(?nonvar,?integer)` - `zero_or_more`

---

increment_counter/1

Increments the named counter.

Compilation flags:
`static`

Template:
`increment_counter(Counter)`
Mode and number of proofs:
`increment_counter(+nonvar)` - `one`

---

decrement_counter/1

Decrements the named counter.

Compilation flags:
`static`

Template:
`decrement_counter(Counter)`
Mode and number of proofs:
`decrement_counter(+nonvar)` - `one`

---

reset_counter/1

Resets the named counter to zero. Creates the counter if it does not exist.

Compilation flags:
`static`

Template:
`reset_counter(Counter)`
Mode and number of proofs:
`reset_counter(+nonvar)` - `one`

---

reset_counters/0

Resets all existing named counters to zero.

Compilation flags:
`static`

Mode and number of proofs:
`reset_counters` - `one`

Protected predicates

(none)

Private predicates

counter_/2

Table of named counters.

Compilation flags:
`dynamic`

Template:
`counter_(Counter,Value)`
Mode and number of proofs:
`counter_(?nonvar,?integer)` - `zero_or_more`

Operators

(none)
object

1.199.3 `streamvars`

Stream variables (supporting logical, backtracable, adding and retrieving of terms).

Availability:
`logtalk_load(library(loader))`

Author: Nobukuni Kino and Paulo Moura
Version: 1:3:0
Date: 2019-06-15

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *new/1*
  - *new/2*
  - *(<=)/2*
  - *(=>)/2*
- *Protected predicates*
- *Private predicates*
- *Operators*
  - *op(100,xfx,<=)*
  - *op(100,xfx,=>)*

Public predicates

`new/1`

Makes `Variable` a stream variable. Initial state will be empty.

Compilation flags:
`static`

Template:

```
new(Variable)
```

Mode and number of proofs:

```
new(--streamvar) - one
```

Exceptions:

Variable is not a variable:

```
type_error(variable,Variable)
```

---

new/2

Makes `Variable` a stream variable and sets its initial state to `Value`.

Compilation flags:

```
static
```

Template:

```
new(Variable,Value)
```

Mode and number of proofs:

```
new(--streamvar,@nonvar) - one
```

Exceptions:

Variable is not a variable:

```
type_error(variable,Variable)
```

(<=)/2

Sets the state of the stream variable `Variable` to `Value` (initializing the variable if needed).

Compilation flags:

```
static
```

Template:

```
Variable<=Value
```

Mode and number of proofs:

```
(?streamvar)<=(@nonvar) - one
```

`(=>)/2`

Unifies `Value` with the current state of the stream variable `Variable`.

Compilation flags:
`static`

Template:
`Variable=>Value`
Mode and number of proofs:
`+streamvar=> ?nonvar - zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

`op(100,xfx,<=)`

Scope:
`public`

`op(100,xfx,=>)`

Scope:
`public`

1.200 linda

object

1.200.1 `linda`

Linda tuple-space implementation for process communication. Provides a server that acts as a shared blackboard where clients can write (`out/1-2`), read (`rd/1-2`), and remove (`in/1-2`) tuples. Uses threaded engines for the server implementation and the sockets library for network communication.

Availability:
`logtalk_load(linda(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-03-27

Compilation flags:

`static, context_switching_calls, threaded`

Imports:

`public` *linda_server*
`public` *linda_client*

Remarks:

- Supported backends: SWI-Prolog, Trealla Prolog, and XVM (requires both multi-threading and sockets support).
- Linda operations: The basic operations are `out/1-2` (write tuple), `in/1-2` (remove tuple, blocking), `rd/1-2` (read tuple, blocking), `in_noblock/1-2` (remove tuple, non-blocking), and `rd_noblock/1-2` (read tuple, non-blocking).
- Tuple matching: Tuples are matched using unification.
- Blocking behavior: The `in/1-2` and `rd/1-2` predicates block until a matching tuple is available. The `in_noblock/1-2` and `rd_noblock/1-2` predicates fail immediately if no matching tuple is found.
- Multiple clients: Multiple clients can connect to the same server. A tuple removed by the `in/1-2` or `in_noblock/1-2` predicates is only removed for one client.
- Multiple servers: A client can connect to multiple servers. The first server it connects to, uses by default the `blackboard` alias.
- API compatibility: The API is inspired by the SICStus Prolog Linda library.
- Network communication: Uses TCP sockets for client-server communication, allowing processes to run on different machines.

Inherited public predicates:

check_option/1 check_options/1 close_client/1 default_option/1 default_options/1 findall_in_noblock/3 findall_in_noblock/4 findall_rd_noblock/3 findall_rd_noblock/4 in/1 in/2 in_list/2 in_list/3 in_noblock/1 in_noblock/2 linda/0 linda/1 linda_client/1 linda_client/2 linda_timeout/2 option/2 option/3 out/1 out/2 rd/1 rd/2 rd_list/2 rd_list/3 rd_noblock/1 rd_noblock/2 shutdown_server/1 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.200.2 `linda_client`

Linda client predicates and client-side connection state. Import into a threaded object together with the linda_server category.

Availability:
`logtalk_load(linda(loader))`

Author: Paulo Moura
Version: 2:0:2
Date: 2026-06-13

Compilation flags:
`static`

Extends:
`public` *options*
Uses:
socket
timeout

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *linda_client/2*
  - *linda_client/1*
  - *close_client/1*
  - *shutdown_server/1*
  - *linda_timeout/2*
  - *out/2*
  - *out/1*
  - *in/2*
  - *in/1*
  - *in_noblock/2*
  - *in_noblock/1*
  - *in_list/3*

      - *in_list/2*
      - *rd/2*
      - *rd/1*
      - *rd_noblock/2*
      - *rd_noblock/1*
      - *rd_list/3*
      - *rd_list/2*
      - *findall_rd_noblock/4*
      - *findall_rd_noblock/3*
      - *findall_in_noblock/4*
      - *findall_in_noblock/3*
    - *Protected predicates*
    - *Private predicates*
      - *client_connection_input_/2*
      - *client_connection_output_/2*
      - *client_connection_alias_/2*
      - *client_timeout_/1*
    - *Operators*

Public predicates

`linda_client/2`

Connects to a Linda server at the given address (`Host:Port`) using the given options.

Compilation flags:
`static`

Template:
`linda_client(Address,Options)`

Mode and number of proofs:
`linda_client(++address,++list(compound))` - `one_or_error`

Exceptions:
Already connected:
`linda_error(already_connected)`
Connection failed:
`linda_error(connection_failed(Error))`

Remarks:

- Option `alias(Alias)`: Use `Alias` as an alias to the server address. Default is `blackboard`.

linda_client/1

Connects to a Linda server at the given address (`Host:Port`) using default options.

Compilation flags:
`static`

Template:
`linda_client(Address)`
Mode and number of proofs:
`linda_client(++address)` - `one_or_error`

Exceptions:
Already connected:
`linda_error(already_connected)`
Connection failed:
`linda_error(connection_failed(Error))`

---

close_client/1

Closes the connection to the given Linda server.

Compilation flags:
`static`

Template:
`close_client(Address)`
Mode and number of proofs:
`close_client(++address_or_alias)` - `one`

---

shutdown_server/1

Sends a shutdown signal to the given server. The server stops accepting new connections but continues serving existing clients until they all disconnect. The connection used for shutdown is closed after the server acknowledges the request.

Compilation flags:
`static`

Template:
`shutdown_server(Address)`
Mode and number of proofs:
`shutdown_server(++address_or_alias)` - `one_or_error`

Exceptions:
Not connected:

```
linda_error(not_connected(Address))
```

---

linda_timeout/2

Gets or sets the client timeout. `OldTime` is unified with the current timeout and the timeout is set to `NewTime`. The timeout value is either `off` (no timeout, wait forever) or `Seconds:Milliseconds`.

Compilation flags:

```
static
```

Template:

```
linda_timeout(OldTime,NewTime)
```

Mode and number of proofs:

```
linda_timeout(?compound,+compound) - one
```

---

out/2

Places the tuple `Tuple` in the tuple-space of the given server.

Compilation flags:

```
static
```

Template:

```
out(AddressOrALias,Tuple)
```

Mode and number of proofs:

```
out(++address_or_alias,+term) - one
```

---

out/1

Places the tuple `Tuple` in the tuple-space of the default server.

Compilation flags:

```
static
```

Template:

```
out(Tuple)
```

Mode and number of proofs:

```
out(+term) - one
```

---

in/2

Removes a tuple matching `Tuple` from the tuple-space of the given server. Blocks if no matching tuple is available.

Compilation flags:
`static`

Template:
`in(AddressOrALias,Tuple)`
Mode and number of proofs:
`in(++address_or_alias,?term) - one`

---

in/1

Removes a tuple matching `Tuple` from the tuple-space of the default server. Blocks if no matching tuple is available.

Compilation flags:
`static`

Template:
`in(Tuple)`
Mode and number of proofs:
`in(?term) - one`

---

in_noblock/2

Removes a tuple matching `Tuple` from the tuple-space of the default server. Fails if no matching tuple is available.

Compilation flags:
`static`

Template:
`in_noblock(AddressOrALias,Tuple)`
Mode and number of proofs:
`in_noblock(++address_or_alias,?term) - zero_or_one`

---

in_noblock/1

Removes a tuple matching `Tuple` from the tuple-space of the default server. Fails if no matching tuple is available.

Compilation flags:
`static`

Template:
`in_noblock(Tuple)`
Mode and number of proofs:
`in_noblock(?term)` - `zero_or_one`

---

in_list/3

Removes a tuple matching one of the patterns in `TupleList` from the tuple-space of the given server. `Tuple` is unified with the matched tuple. Blocks if no matching tuple is available.

Compilation flags:
`static`

Template:
`in_list(AddressOrALias,TupleList,Tuple)`
Mode and number of proofs:
`in_list(++address_or_alias,+list,?term)` - `one`

---

in_list/2

Removes a tuple matching one of the patterns in `TupleList` from the tuple-space of the default server. `Tuple` is unified with the matched tuple. Blocks if no matching tuple is available.

Compilation flags:
`static`

Template:
`in_list(TupleList,Tuple)`
Mode and number of proofs:
`in_list(+list,?term)` - `one`

---

rd/2

Reads a tuple matching `Tuple` from the tuple-space of the given server without removing it. Blocks if no matching tuple is available.

Compilation flags:
`static`

Template:
`rd(AddressOrALias,Tuple)`
Mode and number of proofs:
`rd(++address_or_alias,?term)` - `one`

---

rd/1

Reads a tuple matching `Tuple` from the tuple-space of the default server without removing it. Blocks if no matching tuple is available.

Compilation flags:
`static`

Template:
`rd(Tuple)`
Mode and number of proofs:
`rd(?term)` - `one`

---

rd_noblock/2

Reads a tuple matching `Tuple` from the tuple-space of the given server without removing it. Fails if no matching tuple is available.

Compilation flags:
`static`

Template:
`rd_noblock(AddressOrALias,Tuple)`
Mode and number of proofs:
`rd_noblock(++address_or_alias,?term)` - `zero_or_one`

---

rd_noblock/1

Reads a tuple matching `Tuple` from the tuple-space of the default server without removing it. Fails if no matching tuple is available.

Compilation flags:
`static`

Template:
`rd_noblock(Tuple)`
Mode and number of proofs:
`rd_noblock(?term)` - `zero_or_one`

---

rd_list/3

Reads a tuple matching one of the patterns in `TupleList` from the tuple-space of the given server without removing it. `Tuple` is unified with the matched tuple. Blocks if no matching tuple is available.

Compilation flags:
`static`

Template:
`rd_list(AddressOrALias,TupleList,Tuple)`
Mode and number of proofs:
`rd_list(++address_or_alias,+list,?term)` - `one`

---

rd_list/2

Reads a tuple matching one of the patterns in `TupleList` from the tuple-space of the default server without removing it. `Tuple` is unified with the matched tuple. Blocks if no matching tuple is available.

Compilation flags:
`static`

Template:
`rd_list(TupleList,Tuple)`
Mode and number of proofs:
`rd_list(+list,?term)` - `one`

---

findall_rd_noblock/4

Returns a list of all instances of `Template` for tuples matching `Tuple` in the tuple-space. The operation is atomic.

Compilation flags:
`static`

Template:
`findall_rd_noblock(AddressOrALias,Template,Tuple,List)`
Mode and number of proofs:
`findall_rd_noblock(++address_or_alias,?term,+term,?list) - one`

---

findall_rd_noblock/3

Returns a list of all instances of `Template` for tuples matching `Tuple` in the tuple-space. The operation is atomic.

Compilation flags:
`static`

Template:
`findall_rd_noblock(Template,Tuple,List)`
Mode and number of proofs:
`findall_rd_noblock(?term,+term,?list) - one`

---

findall_in_noblock/4

Removes and returns a list of all instances of `Template` for tuples matching `Tuple` in the tuple-space. The operation is atomic - all matching tuples are removed in one synchronized operation.

Compilation flags:
`static`

Template:
`findall_in_noblock(AddressOrALias,Template,Tuple,List)`
Mode and number of proofs:
`findall_in_noblock(++address_or_alias,?term,+term,?list) - one`

---

findall_in_noblock/3

Removes and returns a list of all instances of `Template` for tuples matching `Tuple` in the tuple-space. The operation is atomic - all matching tuples are removed in one synchronized operation.

Compilation flags:

```
static
```

Template:

```
findall_in_noblock(Template,Tuple,List)
```

Mode and number of proofs:

```
findall_in_noblock(?term,+term,?list) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

client_connection_input_/2

Stores the input stream for the client connection to the server.

Compilation flags:

```
dynamic
```

Template:

```
client_connection_input_(Address,InputStream)
```

Mode and number of proofs:

```
client_connection_input_(?address,?stream) - zero_or_more
```

---

client_connection_output_/2

Stores the output stream for the client connection to the server.

Compilation flags:

```
dynamic
```

Template:

```
client_connection_output_(Address,OutputStream)
```

Mode and number of proofs:

```
client_connection_output_(?address,?stream) - zero_or_more
```

---

client_connection_alias_/2

Stores the client connection server aliases.

Compilation flags:
`dynamic`

Template:
`client_connection_alias_(Address,Alias)`
Mode and number of proofs:
`client_connection_alias_(?address,?atom)` - `zero_or_more`

---

client_timeout_/1

Stores the timeout value for blocking client operations. Value is either `off` or `Seconds:Milliseconds`.

Compilation flags:
`dynamic`

Template:
`client_timeout_(Timeout)`
Mode and number of proofs:
`client_timeout_(?compound)` - `zero_or_one`

Operators

(none)
category

1.200.3 linda_server

Linda server predicates and tuple-space state. Import into a threaded object together with the linda_client category.

Availability:
`logtalk_load(linda(loader))`

Author: Paulo Moura
Version: 2:0:4
Date: 2026-07-14

Compilation flags:
`static`

Provides:
logtalk::message_prefix_stream/4

Uses:

list
logtalk
os
socket

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - `linda/1`
  - `linda/0`
- *Protected predicates*
- *Private predicates*
  - `server_socket_/1`
  - `client_connection_/3`
  - `accept_hook_/1`
  - `server_running_/1`
  - `server_shutdown_/0`
  - `accept_loop_stopped_/0`
  - `tuple_/1`
  - `waiting_/3`
  - `client_engine_/2`
  - `ts_out/1`
  - `ts_in/4`
  - `ts_in_noblock/2`
  - `ts_in_list/4`
  - `ts_rd/4`
  - `ts_rd_noblock/2`
  - `ts_rd_list/4`
  - `ts_findall_rd_noblock/3`
  - `ts_findall_in_noblock/3`
- *Operators*

Public predicates

`linda/1`

Starts a Linda server with the given options. The predicate succeeds when all clients have disconnected after a shutdown request.

Compilation flags:

`static`

Template:

`linda(Options)`

Meta-predicate template:

`linda(::)`

Mode and number of proofs:

`linda(+list)` - `one`

Remarks:

- Option `port(Port)`: Use `Port` as the server port. Must be an integer and an available port.
- Option `Address-Goal`: `Address` is unified with `Host:Port` and `Goal` is called when the server starts. Useful for saving the address or starting clients.
- Option `accept_hook(Client,Input,Output,Goal)`: When a client connects, `Client` is unified with the client address, `Input` and `Output` with the connection streams, and `Goal` is called. If `Goal` fails, the connection is rejected.

`linda/0`

Starts a Linda server on an automatically assigned port using default options. The server address (`Host:Port`) is written to the current output stream. The predicate succeeds when all clients have disconnected after a shutdown request.

Compilation flags:

`static`

Mode and number of proofs:

`linda` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`server_socket_/1`

Stores the server socket descriptor.

Compilation flags:

`dynamic`

Template:

`server_socket_(ServerSocket)`

Mode and number of proofs:

`server_socket_(?term)` - `zero_or_one`

---

client_connection_/3

Stores active client connections. Each client has an ID, input stream, and output stream.

Compilation flags:
`dynamic`

Template:
`client_connection_(ClientId,InputStream,OutputStream)`
Mode and number of proofs:
`client_connection_(?term,?term,?term)` - `zero_or_more`

---

accept_hook_/1

Stores the optional accept hook goal to call when a client connects.

Compilation flags:
`dynamic`

Template:
`accept_hook_(Hook)`
Mode and number of proofs:
`accept_hook_(?callable)` - `zero_or_one`

---

server_running_/1

Flag indicating the server is running.

Compilation flags:
`dynamic`

Template:
`server_running_(Address)`
Mode and number of proofs:
`server_running_(?address)` - `zero_or_one`

---

server_shutdown_/0

Flag indicating the server has received a shutdown request.

Compilation flags:
`dynamic`

Mode and number of proofs:
`server_shutdown_` - `zero_or_one`

---

accept_loop_stopped_/0

Flag indicating the server accept loop has stopped.

Compilation flags:
`dynamic`

Mode and number of proofs:
`accept_loop_stopped_` - `zero_or_one`

---

tuple_/1

Stores tuples in the Linda tuple space.

Compilation flags:
`dynamic`

Template:
`tuple_(Tuple)`
Mode and number of proofs:
`tuple_(?term)` - `zero_or_more`

---

waiting_/3

Stores blocked clients waiting for tuples. Records the client ID, request pattern, and output stream.

Compilation flags:
`dynamic`

Template:
`waiting_(ClientId,Request,OutputStream)`
Mode and number of proofs:
`waiting_(?term,?term,?term)` - `zero_or_more`

---

client_engine_/2

Maps client IDs to their corresponding threaded engine names.

Compilation flags:
```
dynamic
```

Template:
```
client_engine_(ClientId,EngineName)
```
Mode and number of proofs:
```
client_engine_(?term,?atom) - zero_or_more
```

---

ts_out/1

Synchronized predicate to add a tuple to the tuple space and wake waiting clients.

Compilation flags:
```
static, synchronized
```

Template:
```
ts_out(Tuple)
```
Mode and number of proofs:
```
ts_out(+term) - one
```

---

ts_in/4

Synchronized predicate to remove a matching tuple or register a waiting client.

Compilation flags:
```
static, synchronized
```

Template:
```
ts_in(Tuple,ClientId,OutputStream,Found)
```
Mode and number of proofs:
```
ts_in(+term,+term,+term,-compound) - one
```

---

ts_in_noblock/2

Synchronized predicate to try removing a matching tuple without blocking.

Compilation flags:
```
static, synchronized
```

Template:
```
ts_in_noblock(Tuple,Found)
```
Mode and number of proofs:
```
ts_in_noblock(+term,-compound) - zero_or_one
```

---

ts_in_list/4

Synchronized predicate to remove a tuple matching one of multiple patterns or register a waiting client.

Compilation flags:
```
static, synchronized
```

Template:
```
ts_in_list(TupleList,ClientId,OutputStream,Found)
```
Mode and number of proofs:
```
ts_in_list(+list,+term,+term,-compound) - one
```

---

ts_rd/4

Synchronized predicate to read a matching tuple or register a waiting client.

Compilation flags:
```
static, synchronized
```

Template:
```
ts_rd(Tuple,ClientId,OutputStream,Found)
```
Mode and number of proofs:
```
ts_rd(+term,+term,+term,-compound) - one
```

---

ts_rd_noblock/2

Synchronized predicate to try reading a matching tuple without blocking.

Compilation flags:
```
static, synchronized
```

Template:
```
ts_rd_noblock(Tuple,Found)
```
Mode and number of proofs:
```
ts_rd_noblock(+term,-compound) - zero_or_one
```

---

ts_rd_list/4

Synchronized predicate to read a tuple matching one of multiple patterns or register a waiting client.

Compilation flags:
```
static, synchronized
```

Template:
```
ts_rd_list(TupleList,ClientId,OutputStream,Found)
```
Mode and number of proofs:
```
ts_rd_list(+list,+term,+term,-compound) - one
```

---

ts_findall_rd_noblock/3

Synchronized predicate to collect all tuples matching a pattern.

Compilation flags:
```
static, synchronized
```

Template:
```
ts_findall_rd_noblock(Template,Tuple,List)
```
Mode and number of proofs:
```
ts_findall_rd_noblock(+term,+term,-list) - zero_or_more
```

---

ts_findall_in_noblock/3

Synchronized predicate to collect and remove all tuples matching a pattern.

Compilation flags:
`static, synchronized`

Template:
`ts_findall_in_noblock(Template,Tuple,List)`
Mode and number of proofs:
`ts_findall_in_noblock(+term,+term,-list)` - `zero_or_more`

Operators

(none)

1.201 linear_algebra

object

1.201.1 linear_algebra

Linear algebra predicates for numeric vectors and matrices implemented without dependencies on machine learning libraries.

Availability:
`logtalk_load(linear_algebra(loader))`

Author: Paulo Moura
Version: 2:2:0
Date: 2026-10-21

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *linear_algebra_protocol*
Uses:
list
numberlist
type

Remarks:
(none)

Inherited public predicates:

add_matrices/3 *add_scaled_outer_product/4* *add_scaled_vector/4* *add_vectors/3* *backward_substitution/3* *basis_vector/3* *center_gram_matrix/4* *center_gram_vector/4* *cholesky_decomposition/2* *covariance_matrix/2* *determinant/2* *diagonal_matrix/2* *diagonal_matrix/3* *diagonal_matrix/4* *difference_norm/3* *dot_product/3* *euclidean_norm/2* *first_significant_component/2* *first_significant_component/3* *forward_substitution/3* *frobenius_norm/2* *gram_matrix/2* *identity_matrix/2* *inverse_matrix/2* *invert_from_cholesky/2* *least_squares/3* *lower_triangular_part/2* *lower_triangular_part/3* *matrix_column/3* *matrix_column_means/2* *matrix_diagonal/2* *matrix_diagonal/3* *matrix_matrix_product/3* *matrix_rank/2* *matrix_rank/3* *matrix_row/3* *matrix_row_means/2* *matrix_trace/2* *matrix_value/4* *matrix_vector_product/3* *new_matrix/4* *new_vector/3* *new_vector_like/2* *normalize_vector/2* *normalize_vector/3* *null_space/2* *null_space/3* *outer_product/3* *pseudo_inverse/2* *pseudo_inverse/3* *qr_decomposition/3* *scale_matrix/3* *scale_vector/3* *shift_matrix_diagonal/3* *solve_cholesky/3* *solve_linear_system/3* *solve_linear_systems/3* *solve_lower_triangular_matrix/3* *solve_upper_triangular/3* *solve_upper_triangular_matrix/3* *stabilize_vector_sign/2* *stabilize_vector_sign/3* *subtract_matrices/3* *subtract_vectors/3* *symmetric_eigen/3* *symmetric_eigen/4* *symmetric_eigen/5* *symmetric_eigen_partial/7* *transpose_matrix/2* *upper_triangular_part/2* *upper_triangular_part/3* *vector_norm/3* *vector_value/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.201.2 `linear_algebra_protocol`

Linear algebra protocol.

Availability:

```
logtalk_load(linear_algebra(loader))
```

Author: Paulo Moura
Version: 2:1:0
Date: 2026-10-12

Compilation flags:

```
static
```

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *new_vector/3*
  - *new_vector_like/2*
  - *add_vectors/3*
  - *subtract_vectors/3*
  - *add_scaled_vector/4*
  - *scale_vector/3*
  - *dot_product/3*
  - *euclidean_norm/2*
  - *vector_norm/3*
  - *basis_vector/3*
  - *new_matrix/4*
  - *identity_matrix/2*
  - *matrix_vector_product/3*
  - *matrix_matrix_product/3*
  - *gram_matrix/2*
  - *matrix_row_means/2*
  - *matrix_column_means/2*
  - *center_gram_matrix/4*
  - *center_gram_vector/4*
  - *outer_product/3*
  - *add_matrices/3*
  - *subtract_matrices/3*
  - *scale_matrix/3*
  - *frobenius_norm/2*
  - *transpose_matrix/2*
  - *vector_value/3*
  - *matrix_row/3*
  - *matrix_column/3*
  - *matrix_value/4*
  - *matrix_diagonal/2*
  - *matrix_diagonal/3*
  - *diagonal_matrix/2*
  - *diagonal_matrix/3*
  - *diagonal_matrix/4*
  - *matrix_trace/2*
  - *shift_matrix_diagonal/3*
  - *upper_triangular_part/2*

- *upper_triangular_part/3*
- *lower_triangular_part/2*
- *lower_triangular_part/3*
- *solve_linear_system/3*
- *solve_linear_systems/3*
- *determinant/2*
- *inverse_matrix/2*
- *qr_decomposition/3*
- *least_squares/3*
- *matrix_rank/2*
- *matrix_rank/3*
- *symmetric_eigen/3*
- *symmetric_eigen/4*
- *symmetric_eigen/5*
- *symmetric_eigen_partial/7*
- *pseudo_inverse/2*
- *pseudo_inverse/3*
- *null_space/2*
- *null_space/3*
- *normalize_vector/2*
- *normalize_vector/3*
- *difference_norm/3*
- *stabilize_vector_sign/2*
- *stabilize_vector_sign/3*
- *first_significant_component/2*
- *first_significant_component/3*
- *add_scaled_outer_product/4*
- *covariance_matrix/2*
- *cholesky_decomposition/2*
- *solve_cholesky/3*
- *forward_substitution/3*
- *solve_lower_triangular_matrix/3*
- *solve_upper_triangular/3*
- *solve_upper_triangular_matrix/3*
- *backward_substitution/3*
- *invert_from_cholesky/2*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new_vector/3

Constructs a numeric vector with the requested length, filled with the given value.

Compilation flags:
`static`

Template:
`new_vector(Count,Value,Vector)`
Mode and number of proofs:
`new_vector(+integer,+number,-list(number))` - `one`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Value` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Count` is an integer that is less than zero:
`domain_error(non_negative_integer,Count)`

---

new_vector_like/2

Constructs a numeric zero vector matching the length of the reference list.

Compilation flags:
`static`

Template:
`new_vector_like(Reference,Zeroes)`
Mode and number of proofs:
`new_vector_like(+list,-list(number))` - `one`

---

add_vectors/3

Adds two numeric vectors element-wise.

Compilation flags:
`static`

Template:
`add_vectors(Vector1,Vector2,Vector)`
Mode and number of proofs:
`add_vectors(+list(number),+list(number),-list(number))` - `one`

---

subtract_vectors/3

Subtracts the second numeric vector from the first element-wise.

Compilation flags:
`static`

Template:
`subtract_vectors(Vector1,Vector2,Vector)`
Mode and number of proofs:
`subtract_vectors(+list(number),+list(number),-list(number))` - `one`

---

add_scaled_vector/4

Adds a scaled numeric vector to another numeric vector.

Compilation flags:
`static`

Template:
`add_scaled_vector(Vector,Scale,Vector0,UpdatedVector)`
Mode and number of proofs:
`add_scaled_vector(+list(number),+number,+list(number),-list(number))` - `one`

Exceptions:
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`

---

scale_vector/3

Scales a numeric vector by the given factor.

Compilation flags:
`static`

Template:
`scale_vector(Vector,Scale,ScaledVector)`
Mode and number of proofs:
`scale_vector(+list(number),+number,-list(number))` - `one`

Exceptions:
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`

---

dot_product/3

Computes the dot product of two numeric vectors of the same length. Returns the floating-point identity 0.0 for two empty vectors. Fails when the vectors have different lengths.

Compilation flags:
`static`

Template:
`dot_product(Vector1,Vector2,Product)`
Mode and number of proofs:
`dot_product(+list(number),+list(number),-number)` - `zero_or_one`

---

euclidean_norm/2

Computes the Euclidean norm of a numeric vector. Returns the floating-point identity 0.0 for an empty vector.

Compilation flags:
`static`

Template:
`euclidean_norm(Vector,Norm)`
Mode and number of proofs:
`euclidean_norm(+list(number),-float)` - `one`

---

vector_norm/3

Computes the norm of a numeric vector for a positive numeric order or the atoms *inf* and *infinity*. Returns the floating-point identity 0.0 for an empty vector and a valid order.

Compilation flags:
`static`

Template:
`vector_norm(Vector,Order,Norm)`
Mode and number of proofs:
`vector_norm(+list(number),@term,-number)` - `one_or_error`

Exceptions:
`Order` is a variable:
`instantiation_error`
`Order` is neither a variable, a number, nor one of the atoms `inf` and `infinity`:
`type_error(number,Order)`
`Order` is a number but not a valid positive_number_or_infinity:
`domain_error(positive_number_or_infinity,Order)`

---

basis_vector/3

Constructs a numeric basis vector with value 1.0 at the given one-based index and 0.0 elsewhere.

Compilation flags:
`static`

Template:
`basis_vector(Size,Index,Vector)`
Mode and number of proofs:
`basis_vector(+integer,+integer,-list(number))` - `one`

Exceptions:
`Size` is a variable:
`instantiation_error`
`Index` is a variable:
`instantiation_error`
`Size` is neither a variable nor an integer:
`type_error(integer,Size)`
`Index` is neither a variable nor an integer:
`type_error(integer,Index)`
`Size` is an integer that is less than one:
`domain_error(positive_integer,Size)`
`Index` is an integer that is less than one:
`domain_error(positive_integer,Index)`

---

new_matrix/4

Constructs a numeric matrix with the requested row and column counts, filled with the given value.

Compilation flags:
`static`

Template:
`new_matrix(Rows,Columns,Value,Matrix)`
Mode and number of proofs:
`new_matrix(+integer,+integer,+number,-list(list(number)))` - `one`

Exceptions:
`Rows` is a variable:
`instantiation_error`
`Columns` is a variable:
`instantiation_error`
`Value` is a variable:
`instantiation_error`
`Rows` is neither a variable nor an integer:
`type_error(integer,Rows)`
`Columns` is neither a variable nor an integer:
`type_error(integer,Columns)`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Rows` is an integer that is less than zero:
`domain_error(non_negative_integer,Rows)`
`Columns` is an integer that is less than zero:
`domain_error(non_negative_integer,Columns)`

---

identity_matrix/2

Constructs a square identity matrix with the requested size.

Compilation flags:
`static`

Template:
`identity_matrix(Size,Matrix)`
Mode and number of proofs:
`identity_matrix(+integer,-list(list(number)))` - `one`

Exceptions:
`Size` is a variable:
`instantiation_error`
`Size` is neither a variable nor an integer:
`type_error(integer,Size)`

`Size` is an integer that is less than zero:
`domain_error(non_negative_integer,Size)`

---

matrix_vector_product/3

Computes the matrix-vector product for a numeric matrix and vector, returning the list of row-wise dot products between each matrix row and the vector. Zero-width rows multiplied by an empty vector produce 0.0 values. Fails when a row and the vector have different lengths.

Compilation flags:
`static`

Template:
`matrix_vector_product(Matrix,Vector,Product)`
Mode and number of proofs:
`matrix_vector_product(+list(list(number)),+list(number),-list(number))` - `zero_or_one`

---

matrix_matrix_product/3

Computes the matrix-matrix product for two numeric matrices.

Compilation flags:
`static`

Template:
`matrix_matrix_product(Matrix1,Matrix2,Product)`
Mode and number of proofs:
`matrix_matrix_product(+list(list(number)),+list(list(number)),-list(list(number)))` - `one`

---

gram_matrix/2

Computes the row Gram matrix of a numeric row matrix by taking all row-wise dot products. For N zero-width rows, returns an N by N matrix filled with 0.0. Fails when the input rows have different lengths.

Compilation flags:
`static`

Template:
`gram_matrix(Rows,GramMatrix)`
Mode and number of proofs:
`gram_matrix(+list(list(number)),-list(list(number)))` - `zero_or_one`

---

matrix_row_means/2

Computes the arithmetic mean of each row in a numeric matrix.

Compilation flags:
`static`

Template:
`matrix_row_means(Matrix,Means)`
Mode and number of proofs:
`matrix_row_means(+list(list(number)),-list(number))` - `one`

Exceptions:
`Matrix` contains an empty row `Row`:
`domain_error(minimum_number_of_values(1),Row)`

---

matrix_column_means/2

Computes the arithmetic mean of each column in a numeric matrix.

Compilation flags:
`static`

Template:
`matrix_column_means(Matrix,Means)`
Mode and number of proofs:
`matrix_column_means(+list(list(number)),-list(number))` - `one`

---

center_gram_matrix/4

Double-centers a numeric Gram matrix and also returns its row means and total mean.

Compilation flags:
`static`

Template:
`center_gram_matrix(GramMatrix,CenteredGramMatrix,RowMeans,TotalMean)`
Mode and number of proofs:
`center_gram_matrix(+list(list(number)),-list(list(number)),-list(number),-number)` - `one`

Exceptions:
`GramMatrix` is empty:
`domain_error(minimum_number_of_values(1),GramMatrix)`
`GramMatrix` contains an empty row `Row`:

domain_error(minimum_number_of_values(1),Row)

---

center_gram_vector/4

Centers a numeric Gram vector using training row means and total mean.

Compilation flags:
static

Template:
center_gram_vector(GramVector,RowMeans,TotalMean,CenteredGramVector)

Mode and number of proofs:
center_gram_vector(+list(number),+list(number),+number,-list(number)) - one

Exceptions:
GramVector is empty:
domain_error(minimum_number_of_values(1),GramVector)

---

outer_product/3

Computes the outer product of two numeric vectors.

Compilation flags:
static

Template:
outer_product(Vector1,Vector2,Matrix)

Mode and number of proofs:
outer_product(+list(number),+list(number),-list(list(number))) - one

---

add_matrices/3

Adds two numeric matrices element-wise.

Compilation flags:
static

Template:
add_matrices(Matrix1,Matrix2,Matrix)

Mode and number of proofs:
add_matrices(+list(list(number)),+list(list(number)),-list(list(number))) - one

---

subtract_matrices/3

Subtracts the second numeric matrix from the first element-wise.

Compilation flags:
`static`

Template:
`subtract_matrices(Matrix1,Matrix2,Matrix)`
Mode and number of proofs:
`subtract_matrices(+list(list(number)),+list(list(number)),-list(list(number)))` - `one`

---

scale_matrix/3

Scales each element of a numeric matrix by the given factor.

Compilation flags:
`static`

Template:
`scale_matrix(Matrix,Scale,ScaledMatrix)`
Mode and number of proofs:
`scale_matrix(+list(list(number)),+number,-list(list(number)))` - `one`

---

frobenius_norm/2

Computes the Frobenius norm of a numeric matrix.

Compilation flags:
`static`

Template:
`frobenius_norm(Matrix,Norm)`
Mode and number of proofs:
`frobenius_norm(+list(list(number)),-float)` - `one_or_error`

Exceptions:
`Matrix` is not a rectangular matrix:
`domain_error(rectangular_matrix,Matrix)`

---

transpose_matrix/2

Transposes a numeric matrix represented as a list of row lists.

Compilation flags:
`static`

Template:
`transpose_matrix(Matrix,Transpose)`
Mode and number of proofs:
`transpose_matrix(+list(list(number)),-list(list(number)))` - `one`

---

vector_value/3

Looks up a numeric vector element using a one-based index.

Compilation flags:
`static`

Template:
`vector_value(Vector,Index,Value)`
Mode and number of proofs:
`vector_value(+list(number),+integer,-number)` - `one`

Exceptions:
`Index` is a variable:
`instantiation_error`
`Index` is neither a variable nor an integer:
`type_error(integer,Index)`
`Index` is an integer that is less than one:
`domain_error(positive_integer,Index)`

---

matrix_row/3

Looks up a numeric matrix row using a one-based row index.

Compilation flags:
`static`

Template:
`matrix_row(Matrix,RowIndex,Row)`
Mode and number of proofs:
`matrix_row(+list(list(number)),+integer,-list(number))` - `one`

Exceptions:

`RowIndex` is a variable:
`instantiation_error`
`RowIndex` is neither a variable nor an integer:
`type_error(integer,RowIndex)`
`RowIndex` is an integer that is less than one:
`domain_error(positive_integer,RowIndex)`

---

matrix_column/3

Looks up a numeric matrix column using a one-based column index.

Compilation flags:
`static`

Template:
`matrix_column(Matrix,ColumnIndex,Column)`
Mode and number of proofs:
`matrix_column(+list(list(number)),+integer,-list(number))` - `one`

Exceptions:
`ColumnIndex` is a variable:
`instantiation_error`
`ColumnIndex` is neither a variable nor an integer:
`type_error(integer,ColumnIndex)`
`ColumnIndex` is an integer that is less than one:
`domain_error(positive_integer,ColumnIndex)`

---

matrix_value/4

Looks up a numeric matrix element using one-based row and column indices.

Compilation flags:
`static`

Template:
`matrix_value(Matrix,RowIndex,ColumnIndex,Value)`
Mode and number of proofs:
`matrix_value(+list(list(number)),+integer,+integer,-number)` - `one`

Exceptions:
`RowIndex` is a variable:
`instantiation_error`
`ColumnIndex` is a variable:
`instantiation_error`

`RowIndex` is neither a variable nor an integer:
```
type_error(integer,RowIndex)
```
`ColumnIndex` is neither a variable nor an integer:
```
type_error(integer,ColumnIndex)
```
`RowIndex` is an integer that is less than one:
```
domain_error(positive_integer,RowIndex)
```
`ColumnIndex` is an integer that is less than one:
```
domain_error(positive_integer,ColumnIndex)
```

---

matrix_diagonal/2

Extracts the main diagonal of a numeric matrix as a vector.

Compilation flags:
```
static
```

Template:
```
matrix_diagonal(Matrix,Diagonal)
```
Mode and number of proofs:
```
matrix_diagonal(+list(list(number)),-list(number)) - one
```

Exceptions:
`Matrix` is not a rectangular matrix:
```
domain_error(rectangular_matrix,Matrix)
```

---

matrix_diagonal/3

Extracts the diagonal of a numeric matrix at the given offset, where positive offsets select superdiagonals and negative offsets select subdiagonals.

Compilation flags:
```
static
```

Template:
```
matrix_diagonal(Matrix,Offset,Diagonal)
```
Mode and number of proofs:
```
matrix_diagonal(+list(list(number)),+integer,-list(number)) - one_or_error
```

Exceptions:
`Offset` is a variable:
```
instantiation_error
```
`Offset` is neither a variable nor an integer:
```
type_error(integer,Offset)
```
`Matrix` is not a rectangular matrix:
```
domain_error(rectangular_matrix,Matrix)
```

diagonal_matrix/2

Constructs a square diagonal matrix from a numeric vector of diagonal entries.

Compilation flags:
`static`

Template:
`diagonal_matrix(Diagonal,Matrix)`
Mode and number of proofs:
`diagonal_matrix(+list(number),-list(list(number)))` - `one`

diagonal_matrix/3

Constructs the minimal square numeric matrix whose diagonal at the given offset matches the input vector.

Compilation flags:
`static`

Template:
`diagonal_matrix(Diagonal,Offset,Matrix)`
Mode and number of proofs:
`diagonal_matrix(+list(number),+integer,-list(list(number)))` - `one_or_error`

Exceptions:
`Offset` is a variable:
`instantiation_error`
`Offset` is neither a variable nor an integer:
`type_error(integer,Offset)`

diagonal_matrix/4

Constructs a square numeric matrix of the requested size whose diagonal at the given offset matches the input vector.

Compilation flags:
`static`

Template:
`diagonal_matrix(Diagonal,Offset,Size,Matrix)`
Mode and number of proofs:
`diagonal_matrix(+list(number),+integer,+integer,-list(list(number)))` - `one_or_error`

Exceptions:

`Offset` is a variable:

```
instantiation_error
```

`Offset` is neither a variable nor an integer:

```
type_error(integer,Offset)
```

`Size` is a variable:

```
instantiation_error
```

`Size` is neither a variable nor an integer:

```
type_error(integer,Size)
```

`Size` is an integer that is less than zero:

```
domain_error(non_negative_integer,Size)
```

`Size` is smaller than the minimum size required by `Diagonal` and `Offset`:

```
domain_error(minimum_matrix_size(A),Size)
```

---

matrix_trace/2

Computes the trace of a numeric matrix by summing its main diagonal entries.

Compilation flags:

```
static
```

Template:

```
matrix_trace(Matrix,Trace)
```

Mode and number of proofs:

```
matrix_trace(+list(list(number)),-number) - one
```

Exceptions:

`Matrix` is not a rectangular matrix:

```
domain_error(rectangular_matrix,Matrix)
```

---

shift_matrix_diagonal/3

Shifts the main diagonal of a numeric matrix by the given value.

Compilation flags:

```
static
```

Template:

```
shift_matrix_diagonal(Matrix,Shift,ShiftedMatrix)
```

Mode and number of proofs:

```
shift_matrix_diagonal(+list(list(number)),+number,-list(list(number))) - one
```

Exceptions:

`Shift` is a variable:

```
instantiation_error
```

`Shift` is neither a variable nor a number:
```
type_error(number,Shift)
```

---

upper_triangular_part/2

Returns the upper-triangular part of a numeric matrix including the main diagonal.

Compilation flags:
```
static
```

Template:
```
upper_triangular_part(Matrix,UpperTriangular)
```
Mode and number of proofs:
```
upper_triangular_part(+list(list(number)),-list(list(number))) - one_or_error
```

Exceptions:
`Matrix` is not a rectangular matrix:
```
domain_error(rectangular_matrix,Matrix)
```

---

upper_triangular_part/3

Returns the upper-triangular part of a numeric matrix using the given diagonal offset, where positive offsets exclude leading diagonals and negative offsets include additional subdiagonals.

Compilation flags:
```
static
```

Template:
```
upper_triangular_part(Matrix,Offset,UpperTriangular)
```
Mode and number of proofs:
```
upper_triangular_part(+list(list(number)),+integer,-list(list(number))) -
one_or_error
```

Exceptions:
`Offset` is a variable:
```
instantiation_error
```
`Offset` is neither a variable nor an integer:
```
type_error(integer,Offset)
```
`Matrix` is not a rectangular matrix:
```
domain_error(rectangular_matrix,Matrix)
```

---

lower_triangular_part/2

Returns the lower-triangular part of a numeric matrix including the main diagonal.

Compilation flags:

```
static
```

Template:

```
lower_triangular_part(Matrix,LowerTriangular)
```

Mode and number of proofs:

```
lower_triangular_part(+list(list(number)),-list(list(number)))-one_or_error
```

Exceptions:

`Matrix` is not a rectangular matrix:

```
domain_error(rectangular_matrix,Matrix)
```

---

lower_triangular_part/3

Returns the lower-triangular part of a numeric matrix using the given diagonal offset, where positive offsets include additional superdiagonals and negative offsets exclude leading diagonals below the main diagonal.

Compilation flags:

```
static
```

Template:

```
lower_triangular_part(Matrix,Offset,LowerTriangular)
```

Mode and number of proofs:

```
lower_triangular_part(+list(list(number)),+integer,-list(list(number)))-
one_or_error
```

Exceptions:

`Offset` is a variable:

```
instantiation_error
```

`Offset` is neither a variable nor an integer:

```
type_error(integer,Offset)
```

`Matrix` is not a rectangular matrix:

```
domain_error(rectangular_matrix,Matrix)
```

---

solve_linear_system/3

Solves a non-singular square numeric linear system with a vector right-hand side using pivoted elimination.

Compilation flags:
`static`

Template:
`solve_linear_system(Matrix,Values,Solution)`
Mode and number of proofs:
`solve_linear_system(+list(list(number)),+list(number),-list(float))` - `one_or_error`

Exceptions:
`Matrix` is not a square matrix:
`domain_error(square_matrix,Matrix)`
`Values` does not have the same length as the size of `Matrix`:
`domain_error(vector_length(A),Values)`
`Matrix` is singular:
`evaluation_error(zero_divisor)`

---

solve_linear_systems/3

Solves a non-singular square numeric linear system with a matrix right-hand side using pivoted elimination.

Compilation flags:
`static`

Template:
`solve_linear_systems(Matrix,RightHandSides,Solutions)`
Mode and number of proofs:
`solve_linear_systems(+list(list(number)),+list(list(number)),-list(list(float)))` - `one_or_error`

Exceptions:
`Matrix` is not a square matrix:
`domain_error(square_matrix,Matrix)`
`RightHandSides` does not have the same number of rows as the size of `Matrix`:
`domain_error(matrix_row_count(A),RightHandSides)`
`Matrix` is singular:
`evaluation_error(zero_divisor)`

---

determinant/2

Computes the determinant of a square numeric matrix.

Compilation flags:
`static`

Template:
`determinant(Matrix,Determinant)`
Mode and number of proofs:
`determinant(+list(list(number)),-number)` - `one_or_error`

Exceptions:
`Matrix` is not a square matrix:
`domain_error(square_matrix,Matrix)`

---

inverse_matrix/2

Computes the inverse of a non-singular square numeric matrix by solving one linear system per identity column.

Compilation flags:
`static`

Template:
`inverse_matrix(Matrix,Inverse)`
Mode and number of proofs:
`inverse_matrix(+list(list(number)),-list(list(float)))` - `one_or_error`

Exceptions:
`Matrix` is not a square matrix:
`domain_error(square_matrix,Matrix)`
`Matrix` is singular:
`evaluation_error(zero_divisor)`

---

qr_decomposition/3

Computes a thin QR decomposition of a rectangular numeric matrix, returning orthogonal columns and an upper-trapezoidal factor.

Compilation flags:
`static`

Template:
`qr_decomposition(Matrix,Orthogonal,UpperTriangular)`
Mode and number of proofs:

```
qr_decomposition(+list(list(number)),-list(list(float)),-list(list(float))) -
one_or_error
```

Exceptions:

`Matrix` is not a rectangular matrix:

```
domain_error(rectangular_matrix,Matrix)
```

---

least_squares/3

Computes a least-squares solution for a rectangular numeric linear system using a pivoted orthogonal solve without forming normal equations.

Compilation flags:

```
static
```

Template:

```
least_squares(Matrix,Values,Solution)
```

Mode and number of proofs:

```
least_squares(+list(list(number)),+list(number),-list(float)) - one_or_error
```

Exceptions:

`Matrix` is not a rectangular matrix:

```
domain_error(rectangular_matrix,Matrix)
```

`Values` does not have the same length as the number of rows of `Matrix`:

```
domain_error(vector_length(A),Values)
```

---

matrix_rank/2

Estimates the rank of a numeric matrix using the default library numerical tolerance of 1.0e-12.

Compilation flags:

```
static
```

Template:

```
matrix_rank(Matrix,Rank)
```

Mode and number of proofs:

```
matrix_rank(+list(list(number)),-integer) - one_or_error
```

Exceptions:

`Matrix` is not a rectangular matrix:

```
domain_error(rectangular_matrix,Matrix)
```

---

matrix_rank/3

Estimates the rank of a numeric matrix using the given non-negative numerical tolerance.

Compilation flags:

`static`

Template:

`matrix_rank(Matrix,Tolerance,Rank)`

Mode and number of proofs:

`matrix_rank(+list(list(number)),+number,-integer)` - `one_or_error`

Exceptions:

`Tolerance` is a variable:

`instantiation_error`

`Tolerance` is neither a variable nor a number:

`type_error(number,Tolerance)`

`Tolerance` is a number but not a non-negative number:

`domain_error(non_negative_number,Tolerance)`

`Matrix` is not a rectangular matrix:

`domain_error(rectangular_matrix,Matrix)`

---

symmetric_eigen/3

Computes an orthonormal eigendecomposition of a real symmetric dense matrix using the default library numerical tolerance of 1.0e-12.

Compilation flags:

`static`

Template:

`symmetric_eigen(Matrix,Eigenvectors,Eigenvalues)`

Mode and number of proofs:

`symmetric_eigen(+list(list(number)),-list(list(float)),-list(float))` - `one_or_error`

Exceptions:

`Matrix` is not a square matrix:

`domain_error(square_matrix,Matrix)`

`Matrix` is not symmetric within the default numerical tolerance:

`domain_error(symmetric_matrix,Matrix)`

---

symmetric_eigen/4

Computes an orthonormal eigendecomposition of a real symmetric dense matrix using the given non-negative numerical tolerance.

Compilation flags:
`static`

Template:
`symmetric_eigen(Matrix,Tolerance,Eigenvectors,Eigenvalues)`

Mode and number of proofs:
`symmetric_eigen(+list(list(number)),+number,-list(list(float)),-list(float))` - `one_or_error`

Exceptions:
`Tolerance` is a variable:
`instantiation_error`
`Tolerance` is neither a variable nor a number:
`type_error(number,Tolerance)`
`Tolerance` is a number but not a non-negative number:
`domain_error(non_negative_number,Tolerance)`
`Matrix` is not a square matrix:
`domain_error(square_matrix,Matrix)`
`Matrix` is not symmetric within `Tolerance`:
`domain_error(symmetric_matrix,Matrix)`

---

symmetric_eigen/5

Computes an orthonormal eigendecomposition of a real symmetric dense matrix using the given non-negative numerical tolerance and positive iteration bound.

Compilation flags:
`static`

Template:
`symmetric_eigen(Matrix,Tolerance,MaximumIterations,Eigenvectors,Eigenvalues)`

Mode and number of proofs:
`symmetric_eigen(+list(list(number)),+number,+integer,-list(list(float)),-list(float))` - `one_or_error`

Exceptions:
`Tolerance` is a variable:
`instantiation_error`
`Tolerance` is neither a variable nor a number:
`type_error(number,Tolerance)`
`Tolerance` is a number but not a non-negative number:
`domain_error(non_negative_number,Tolerance)`
`MaximumIterations` is a variable:
`instantiation_error`

`MaximumIterations` is neither a variable nor an integer:
`type_error(integer,MaximumIterations)`
`MaximumIterations` is an integer that is less than one:
`domain_error(positive_integer,MaximumIterations)`
`Matrix` is not a square matrix:
`domain_error(square_matrix,Matrix)`
`Matrix` is not symmetric within `Tolerance`:
`domain_error(symmetric_matrix,Matrix)`

---

symmetric_eigen_partial/7

Computes a bounded partial largest-magnitude symmetric eigendecomposition with residual and orthogonality diagnostics. Returns eigenpairs in descending algebraic order.

Compilation flags:
`static`

Template:
`symmetric_eigen_partial(Matrix,Count,Tolerance,MaximumIterations,Eigenvectors,Eigenvalues,Diagnostics)`

Mode and number of proofs:
`symmetric_eigen_partial(+list(list(number)),+integer,+number,+integer,-list(list(float)),-list(float),-list(compound))` - `one_or_error`

Exceptions:
`Count`, `Tolerance`, or `MaximumIterations` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Tolerance` is neither a variable nor a number:
`type_error(number,Tolerance)`
`MaximumIterations` is neither a variable nor an integer:
`type_error(integer,MaximumIterations)`
`Count` is an integer less than zero:
`domain_error(non_negative_integer,Count)`
`Count` exceeds the matrix dimension:
`domain_error(component_count(0,Size),Count)`
`Tolerance` is a negative number:
`domain_error(non_negative_number,Tolerance)`
`MaximumIterations` is an integer less than one:
`domain_error(positive_integer,MaximumIterations)`
`Matrix` is not square:
`domain_error(square_matrix,Matrix)`
`Matrix` is not symmetric within `Tolerance`:
`domain_error(symmetric_matrix,Matrix)`

---

pseudo_inverse/2

Computes the Moore-Penrose pseudo-inverse of a rectangular numeric matrix using the default library numerical tolerance of 1.0e-12.

Compilation flags:
`static`

Template:
`pseudo_inverse(Matrix,PseudoInverse)`
Mode and number of proofs:
`pseudo_inverse(+list(list(number)),-list(list(float)))` - `one_or_error`

Exceptions:
`Matrix` is not a rectangular matrix:
`domain_error(rectangular_matrix,Matrix)`

---

pseudo_inverse/3

Computes the Moore-Penrose pseudo-inverse of a rectangular numeric matrix using the given non-negative numerical tolerance.

Compilation flags:
`static`

Template:
`pseudo_inverse(Matrix,Tolerance,PseudoInverse)`
Mode and number of proofs:
`pseudo_inverse(+list(list(number)),+number,-list(list(float)))` - `one_or_error`

Exceptions:
`Tolerance` is a variable:
`instantiation_error`
`Tolerance` is neither a variable nor a number:
`type_error(number,Tolerance)`
`Tolerance` is a number but not a non-negative number:
`domain_error(non_negative_number,Tolerance)`
`Matrix` is not a rectangular matrix:
`domain_error(rectangular_matrix,Matrix)`

---

null_space/2

Computes an orthonormal basis for the right null space of a rectangular numeric matrix using the default library numerical tolerance of 1.0e-12.

Compilation flags:

```
static
```

Template:

```
null_space(Matrix,Basis)
```

Mode and number of proofs:

```
null_space(+list(list(number)),-list(list(float))) - one_or_error
```

Exceptions:

`Matrix` is not a rectangular matrix:

```
domain_error(rectangular_matrix,Matrix)
```

---

null_space/3

Computes an orthonormal basis for the right null space of a rectangular numeric matrix using the given non-negative numerical tolerance.

Compilation flags:

```
static
```

Template:

```
null_space(Matrix,Tolerance,Basis)
```

Mode and number of proofs:

```
null_space(+list(list(number)),+number,-list(list(float))) - one_or_error
```

Exceptions:

`Tolerance` is a variable:

```
instantiation_error
```

`Tolerance` is neither a variable nor a number:

```
type_error(number,Tolerance)
```

`Tolerance` is a number but not a non-negative number:

```
domain_error(non_negative_number,Tolerance)
```

`Matrix` is not a rectangular matrix:

```
domain_error(rectangular_matrix,Matrix)
```

---

normalize_vector/2

Normalizes a numeric vector to unit length when its norm is above the default library numerical tolerance of 1.0e-12.

Compilation flags:
`static`

Template:
`normalize_vector(Vector,NormalizedVector)`
Mode and number of proofs:
`normalize_vector(+list(number),-list(number))` - `one`

---

normalize_vector/3

Normalizes a numeric vector to unit length when its norm is above the given non-negative numerical tolerance.

Compilation flags:
`static`

Template:
`normalize_vector(Vector,Tolerance,NormalizedVector)`
Mode and number of proofs:
`normalize_vector(+list(number),+number,-list(number))` - `one_or_error`

Exceptions:
`Tolerance` is a variable:
`instantiation_error`
`Tolerance` is neither a variable nor a number:
`type_error(number,Tolerance)`
`Tolerance` is a number but not a non-negative number:
`domain_error(non_negative_number,Tolerance)`

---

difference_norm/3

Computes the Euclidean norm of the difference between two numeric vectors.

Compilation flags:
`static`

Template:
`difference_norm(Vector1,Vector2,Norm)`
Mode and number of proofs:
`difference_norm(+list(number),+list(number),-float)` - `one`

---

stabilize_vector_sign/2

Normalizes a vector sign convention by flipping vectors whose first significant component is negative.

Compilation flags:
`static`

Template:
`stabilize_vector_sign(Vector,StableVector)`
Mode and number of proofs:
`stabilize_vector_sign(+list(number),-list(number))` - `one`

---

stabilize_vector_sign/3

Normalizes a vector sign convention by flipping vectors whose first significant component under the given non-negative numerical tolerance is negative.

Compilation flags:
`static`

Template:
`stabilize_vector_sign(Vector,Tolerance,StableVector)`
Mode and number of proofs:
`stabilize_vector_sign(+list(number),+number,-list(number))` - `one_or_error`

Exceptions:
`Tolerance` is a variable:
`instantiation_error`
`Tolerance` is neither a variable nor a number:
`type_error(number,Tolerance)`
`Tolerance` is a number but not a non-negative number:
`domain_error(non_negative_number,Tolerance)`

---

first_significant_component/2

Returns the first component whose absolute value exceeds the default library numerical tolerance of 1.0e-12, defaulting to zero when no such component exists.

Compilation flags:
`static`

Template:
`first_significant_component(Vector,First)`
Mode and number of proofs:
`first_significant_component(+list(number),-number)` - `one`

---

first_significant_component/3

Returns the first component whose absolute value exceeds the given non-negative numerical tolerance, defaulting to zero when no such component exists.

Compilation flags:

```
static
```

Template:

```
first_significant_component(Vector,Tolerance,First)
```

Mode and number of proofs:

```
first_significant_component(+list(number),+number,-number) - one_or_error
```

Exceptions:

`Tolerance` is a variable:

```
instantiation_error
```

`Tolerance` is neither a variable nor a number:

```
type_error(number,Tolerance)
```

`Tolerance` is a number but not a non-negative number:

```
domain_error(non_negative_number,Tolerance)
```

---

add_scaled_outer_product/4

Adds a scaled outer product of a vector with itself to an accumulated matrix.

Compilation flags:

```
static
```

Template:

```
add_scaled_outer_product(Vector,Scale,Matrix0,Matrix)
```

Mode and number of proofs:

```
add_scaled_outer_product(+list(number),+number,+list(list(number)),
-list(list(number))) - one
```

Exceptions:

`Scale` is a variable:

```
instantiation_error
```

`Scale` is neither a variable nor a number:

```
type_error(number,Scale)
```

---

covariance_matrix/2

Computes the sample covariance matrix from a list of centered numeric row vectors.

Compilation flags:
`static`

Template:
`covariance_matrix(Rows,CovarianceMatrix)`
Mode and number of proofs:
`covariance_matrix(+list(list(number)),-list(list(number)))` - `one_or_error`

Exceptions:
`Rows` contains fewer than two rows:
`domain_error(minimum_number_of_rows(2),Rows)`

---

cholesky_decomposition/2

Computes the lower-triangular Cholesky factor of a positive definite numeric matrix.

Compilation flags:
`static`

Template:
`cholesky_decomposition(Matrix,CholeskyFactor)`
Mode and number of proofs:
`cholesky_decomposition(+list(list(number)),-list(list(float)))` - `one_or_error`

Exceptions:
`Matrix` is not numerically positive definite:
`domain_error(positive_definite_matrix,Matrix)`

---

solve_cholesky/3

Solves a positive definite linear system from a lower-triangular Cholesky factor and a right-hand-side vector.

Compilation flags:
`static`

Template:
`solve_cholesky(CholeskyFactor,Values,Solution)`
Mode and number of proofs:
`solve_cholesky(+list(list(number)),+list(number),-list(float))` - `one`

---

forward_substitution/3

Solves a lower-triangular system by forward substitution.

Compilation flags:
```
static
```

Template:
```
forward_substitution(LowerTriangular,Values,Solution)
```
Mode and number of proofs:
```
forward_substitution(+list(list(number)),+list(number),-list(float)) - one
```

---

solve_lower_triangular_matrix/3

Solves a lower-triangular system with a matrix right-hand side by forward substitution on each column.

Compilation flags:
```
static
```

Template:
```
solve_lower_triangular_matrix(LowerTriangular,Matrix,Solution)
```
Mode and number of proofs:
```
solve_lower_triangular_matrix(+list(list(number)),+list(list(number)),
-list(list(float))) - one
```

---

solve_upper_triangular/3

Solves an upper-triangular system by backward substitution.

Compilation flags:
```
static
```

Template:
```
solve_upper_triangular(UpperTriangular,Values,Solution)
```
Mode and number of proofs:
```
solve_upper_triangular(+list(list(number)),+list(number),-list(float)) - one
```

---

solve_upper_triangular_matrix/3

Solves an upper-triangular system with a matrix right-hand side by backward substitution on each column.

Compilation flags:

```
static
```

Template:

```
solve_upper_triangular_matrix(UpperTriangular,Matrix,Solution)
```

Mode and number of proofs:

```
solve_upper_triangular_matrix(+list(list(number)),+list(list(number)),
-list(list(float))) - one
```

---

backward_substitution/3

Solves the transposed upper-triangular system induced by a lower-triangular Cholesky factor using backward substitution.

Compilation flags:

```
static
```

Template:

```
backward_substitution(CholeskyFactor,Values,Solution)
```

Mode and number of proofs:

```
backward_substitution(+list(list(number)),+list(number),-list(float)) - one
```

---

invert_from_cholesky/2

Computes a matrix inverse from a Cholesky factor by solving one linear system per basis column.

Compilation flags:

```
static
```

Template:

```
invert_from_cholesky(CholeskyFactor,Inverse)
```

Mode and number of proofs:

```
invert_from_cholesky(+list(list(number)),-list(list(float))) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.202 linear_programming

category

1.202.1 `linear_programming_common`

Shared immutable model construction, validation, matrix conversion, result inspection, and printing for linear-programming solvers.

Availability:
`logtalk_load(linear_programming(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static`

Implements:
`public` *linear_programming_protocol*
Extends:
`public` *options*
Uses:
format
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *constraint/5* *constraints/3* *default_option/1* *default_options/1* *new_problem/1* *objective/4* *objective_value/2* *option/2* *option/3* *print_problem/1* *print_solution/1* *problem_from_matrices/8* *solve/2* *solve/3* *statistics/2* *status/2* *valid_option/1* *valid_options/1* *variable/4* *variable/6* *variable_value/3*

- *Public predicates*
- *Protected predicates*
  - *check_problem/1*

    - *check_result/1*
    - *relax_problem/2*
    - *discrete_variables/2*
    - *tighten_variable_bounds/5*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

check_problem/1

Checks that the argument is a structurally valid linear-program problem.

Compilation flags:
`static`

Template:
`check_problem(Problem)`
Mode and number of proofs:
`check_problem(@term)` - `one_or_error`

Exceptions:
`Problem` is a variable:
`instantiation_error`
`Problem` is not a linear-program problem:
`type_error(linear_program,Problem)`

---

check_result/1

Checks that the argument is a structurally valid linear-programming result.

Compilation flags:
`static`

Template:
`check_result(Result)`
Mode and number of proofs:
`check_result(@term)` - `one_or_error`

Exceptions:
`Result` is a variable:
`instantiation_error`

`Result` is not a linear-programming result:
`type_error(linear_programming_result,Result)`

---

relax_problem/2

Returns the continuous relaxation of a validated linear-program problem.

Compilation flags:
`static`

Template:
`relax_problem(Problem,Relaxation)`
Mode and number of proofs:
`relax_problem(+compound,-compound)` - `one`

---

discrete_variables/2

Returns the integer and binary variable declarations of a validated linear-program problem in declaration order.

Compilation flags:
`static`

Template:
`discrete_variables(Problem,Variables)`
Mode and number of proofs:
`discrete_variables(+compound,-list(compound))` - `one`

---

tighten_variable_bounds/5

Returns a copy of a validated problem with the named variable bounds tightened to the given finite bounds.

Compilation flags:
`static`

Template:
`tighten_variable_bounds(Name,Lower,Upper,Problem0,Problem)`
Mode and number of proofs:
`tighten_variable_bounds(+term,+number,+number,+compound,-compound)` - `one_or_error`

Exceptions:
An argument is insufficiently instantiated:
`instantiation_error`

`Problem0` is not a linear-program problem:
```
type_error(linear_program,Problem0)
```
`Name` is not a declared variable:
```
domain_error(linear_programming_variable,Name)
```
The bounds are invalid, inconsistent, or weaken the current bounds:
```
domain_error(linear_programming_bounds,Lower-Upper)
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.202.2 `linear_programming_protocol`

Protocol for immutable linear-program construction, solving, and result inspection.

Availability:
```
logtalk_load(linear_programming(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
```
static
```

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *new_problem/1*
  - *variable/4*
  - *variable/6*
  - *constraint/5*
  - *constraints/3*

- *objective/4*
- *problem_from_matrices/8*
- *solve/2*
- *solve/3*
- *status/2*
- *objective_value/2*
- *variable_value/3*
- *statistics/2*
- *print_problem/1*
- *print_solution/1*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new_problem/1

Creates an empty linear-program problem.

Compilation flags:
`static`

Template:
`new_problem(Problem)`

Mode and number of proofs:
`new_problem(-compound)` - `one`

---

variable/4

Adds a variable with default bounds. Continuous and integer variables default to zero and positive infinity; binary variables default to zero and one. The variable type is `continuous`, `integer`, or `binary`.

Compilation flags:
`static`

Template:
`variable(Name,Type,Problem0,Problem)`

Mode and number of proofs:
`variable(+term,+atom,+compound,-compound)` - `one_or_error`

Exceptions:
An argument is insufficiently instantiated:
`instantiation_error`
`Problem0` is not a linear-program problem:
`type_error(linear_program,Problem0)`

`Type` is not a supported variable type:
`domain_error(linear_programming_variable_type,Type)`
`Name` is already declared:
`domain_error(linear_programming_variable,Name)`

---

variable/6

Adds a variable with explicit lower and upper bounds. A bound is a number, `inf`, or `-inf` as appropriate.

Compilation flags:
`static`

Template:
`variable(Name,Type,Lower,Upper,Problem0,Problem)`
Mode and number of proofs:
`variable(+term,+atom,+term,+term,+compound,-compound)` - `one_or_error`

Exceptions:
An argument is insufficiently instantiated:
`instantiation_error`
`Problem0` is not a linear-program problem:
`type_error(linear_program,Problem0)`
`Type` is not a supported variable type:
`domain_error(linear_programming_variable_type,Type)`
`Name` is already declared:
`domain_error(linear_programming_variable,Name)`
The bounds are invalid or inconsistent:
`domain_error(linear_programming_bounds,Lower-Upper)`

---

constraint/5

Adds a linear constraint. `Coefficients` is a list of `Coefficient*Variable` terms and `Sense` is `=<`, `>=`, or `=`.

Compilation flags:
`static`

Template:
`constraint(Coefficients,Sense,RightHandSide,Problem0,Problem)`
Mode and number of proofs:
`constraint(+list(compound),+atom,+number,+compound,-compound)` - `one_or_error`

Exceptions:
An argument is insufficiently instantiated:
`instantiation_error`

`Problem0` is not a linear-program problem:
`type_error(linear_program,Problem0)`
`Coefficients` is not a list of coefficient-variable terms:
`type_error(linear_expression,Coefficients)`
`Sense` is not a supported constraint sense:
`domain_error(linear_programming_constraint_sense,Sense)`
A coefficient references an undeclared variable:
`domain_error(linear_programming_variable,Variable)`

---

constraints/3

Adds a list of `constraint(Coefficients, Sense, RightHandSide)` terms.

Compilation flags:
`static`

Template:
`constraints(Constraints,Problem0,Problem)`
Mode and number of proofs:
`constraints(+list(compound),+compound,-compound)` - `one_or_error`

Exceptions:
An argument is insufficiently instantiated:
`instantiation_error`
`Constraints` is not a list of valid constraint terms:
`type_error(linear_programming_constraints,Constraints)`
`Problem0` is not a linear-program problem:
`type_error(linear_program,Problem0)`

---

objective/4

Sets the linear objective. `Coefficients` is a list of `Coefficient*Variable` terms and `Sense` is `minimize` or `maximize`.

Compilation flags:
`static`

Template:
`objective(Coefficients,Sense,Problem0,Problem)`
Mode and number of proofs:
`objective(+list(compound),+atom,+compound,-compound)` - `one_or_error`

Exceptions:
An argument is insufficiently instantiated:
`instantiation_error`

`Problem0` is not a linear-program problem:
`type_error(linear_program,Problem0)`
`Coefficients` is not a list of coefficient-variable terms:
`type_error(linear_expression,Coefficients)`
`Sense` is not a supported objective sense:
`domain_error(linear_programming_objective_sense,Sense)`
The problem already has an objective:
`domain_error(linear_programming_objective,Problem0)`
A coefficient references an undeclared variable:
`domain_error(linear_programming_variable,Variable)`

---

problem_from_matrices/8

Creates a problem from a dense objective vector, objective sense, equality rows and right-hand sides, less-than-or-equal rows and right-hand sides, and variable bounds. Variables are named by their one-based column indices.

Compilation flags:
`static`

Template:
```
problem_from_matrices(Objective,ObjectiveSense,EqualityMatrix,
EqualityRightHandSide,InequalityMatrix,InequalityRightHandSide,Bounds,Problem)
```
Mode and number of proofs:
```
problem_from_matrices(+list(number),+atom,+list(list(number)),+list(number),
+list(list(number)),+list(number),+list(pair),-compound) - one_or_error
```

Exceptions:
An argument is insufficiently instantiated:
`instantiation_error`
The matrix and vector dimensions are inconsistent:
`domain_error(linear_programming_dimensions,Objective)`
An input is not a numeric vector, numeric matrix, or valid bounds list:
`type_error(linear_programming_matrix_problem,Objective)`

---

solve/2

Solves a linear-program problem using default options and returns a result term. Solver statuses include `optimal`, `infeasible`, `unbounded`, `iteration_limit`, `node_limit`, and `numerical_error` as applicable to the backend.

Compilation flags:
`static`

Template:
`solve(Problem,Result)`
Mode and number of proofs:

`solve(+compound,-compound)` - `one_or_error`

Exceptions:

`Problem` is a variable:
`instantiation_error`
`Problem` is not a linear-program problem:
`type_error(linear_program,Problem)`
`Problem` has no variables:
`domain_error(linear_programming_problem,empty)`
`Problem` has no objective:
`domain_error(linear_programming_problem,missing_objective)`
`Problem` contains a variable type unsupported by the backend:
`domain_error(simplex_variable_type,Variable-Type)`
A discrete variable does not have finite bounds:
`domain_error(milp_finite_integer_bounds,Variable-(Lower-Upper))`
A discrete variable domain contains no integer:
`domain_error(milp_integer_domain,Variable-(Lower-Upper))`

`solve/3`

Solves a linear-program problem using the specified options and returns a result term. Solver statuses include `optimal`, `infeasible`, `unbounded`, `iteration_limit`, `node_limit`, and `numerical_error` as applicable to the backend.

Compilation flags:
`static`

Template:
`solve(Problem,Result,Options)`
Mode and number of proofs:
`solve(+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Problem` is a variable:
`instantiation_error`
`Problem` is not a linear-program problem:
`type_error(linear_program,Problem)`
`Problem` has no variables:
`domain_error(linear_programming_problem,empty)`
`Problem` has no objective:
`domain_error(linear_programming_problem,missing_objective)`
`Problem` contains a variable type unsupported by the backend:
`domain_error(simplex_variable_type,Variable-Type)`
A discrete variable does not have finite bounds:
`domain_error(milp_finite_integer_bounds,Variable-(Lower-Upper))`
A discrete variable domain contains no integer:
`domain_error(milp_integer_domain,Variable-(Lower-Upper))`
`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

---

status/2

Returns the result status.

Compilation flags:

`static`

Template:

`status(Result,Status)`

Mode and number of proofs:

`status(+compound,-atom)` - `one_or_error`

Exceptions:

`Result` is a variable:

`instantiation_error`

`Result` is not a linear-programming result:

`type_error(linear_programming_result,Result)`

---

objective_value/2

Returns the optimal objective value. Fails when the result status is not `optimal`.

Compilation flags:

`static`

Template:

`objective_value(Result,Value)`

Mode and number of proofs:

`objective_value(+compound,-number)` - `zero_or_one_or_error`

Exceptions:

`Result` is a variable:

`instantiation_error`

`Result` is not a linear-programming result:

`type_error(linear_programming_result,Result)`

---

variable_value/3

Returns an optimal variable value. Fails when the result status is not `optimal` or the variable is absent.

Compilation flags:
`static`

Template:
`variable_value(Result,Variable,Value)`
Mode and number of proofs:
`variable_value(+compound,+term,-number)` - `zero_or_one_or_error`

Exceptions:
`Result` is a variable:
`instantiation_error`
`Result` is not a linear-programming result:
`type_error(linear_programming_result,Result)`

---

statistics/2

Returns solver statistics stored in a result.

Compilation flags:
`static`

Template:
`statistics(Result,Statistics)`
Mode and number of proofs:
`statistics(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Result` is a variable:
`instantiation_error`
`Result` is not a linear-programming result:
`type_error(linear_programming_result,Result)`

---

print_problem/1

Prints a linear-program problem to the current output stream.

Compilation flags:
`static`

Template:
`print_problem(Problem)`
Mode and number of proofs:
`print_problem(+compound)` - `one_or_error`

Exceptions:
`Problem` is a variable:
`instantiation_error`
`Problem` is not a linear-program problem:
`type_error(linear_program,Problem)`

---

print_solution/1

Prints a linear-program result to the current output stream.

Compilation flags:
`static`

Template:
`print_solution(Result)`
Mode and number of proofs:
`print_solution(+compound)` - `one_or_error`

Exceptions:
`Result` is a variable:
`instantiation_error`
`Result` is not a linear-programming result:
`type_error(linear_programming_result,Result)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.202.3 `linear_programming_types`

Type definitions for linear-programming model inputs.

Availability:
`logtalk_load(linear_programming(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static`

Provides:
type::type/1
type::check/2

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.202.4 `milp_branch_and_bound`

Portable depth-first branch-and-bound solver for small mixed-integer linear programs.

Availability:
`logtalk_load(linear_programming(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *linear_programming_common*
Uses:
list
simplex

Remarks:

- Discrete domains: Integer and binary variables must have finite bounds. No cutting planes or primal heuristics are used.
- LP relaxations: Each search node is solved using the simplex object.
- Branching: The solver can branch on the first fractional variable or the most fractional variable, with declaration order breaking ties. The lower or upper branch can be explored first.

Inherited public predicates:
check_option/1 check_options/1 constraint/5 constraints/3 default_option/1 default_options/1 new_problem/1 objective/4 objective_value/2 option/2 option/3 print_problem/1 print_solution/1 problem_from_matrices/8 solve/2 solve/3 statistics/2 status/2 valid_option/1 valid_options/1 variable/4 variable/6 variable_value/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

simplex, *linear_programming_protocol*

object

1.202.5 `simplex`

Portable dense two-phase simplex solver for small continuous linear programs.

Availability:
`logtalk_load(linear_programming(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *linear_programming_common*
Uses:
linear_algebra
list

Remarks:

- Variable types: This backend supports continuous variables only. Integer and binary variables are accepted by the shared model API for use by future MILP backends but rejected when solving.
- Pivoting: Bland's rule is used by default. Dantzig's rule is also available, with the lowest column index breaking ties. Minimum-ratio ties use the lowest-indexed basic variable.

Inherited public predicates:
check_option/1 *check_options/1* *constraint/5* *constraints/3* *default_option/1* *default_options/1* *new_problem/1* *objective/4* *objective_value/2* *option/2* *option/3* *print_problem/1* *print_solution/1* *problem_from_matrices/8* *solve/2* *solve/3* *statistics/2* *status/2* *valid_option/1* *valid_options/1* *variable/4* *variable/6* *variable_value/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

linear_programming_protocol

1.203 linear_regression

object

1.203.1 `linear_regression`

Linear regression regressor supporting continuous and mixed-feature datasets using a direct ordinary least-squares solve. Learns from a dataset object implementing the `regression_dataset_protocol` protocol and returns a regressor term that can be used for prediction and exported as predicate clauses.

Availability:
```
logtalk_load(linear_regression(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-13

Compilation flags:
```
static, context_switching_calls
```

Imports:
```
public regressor_common
```
Uses:

format
list
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ridge_regression, *knn_regression*, *regression_tree*, *random_forest_regression*, *gradient_boosting_regression*

1.204 linear_svm_classifier

object

1.204.1 `linear_svm_classifier`

Linear support vector machine classifier supporting binary and multiclass classification using one-vs-rest margin models. Learns from a dataset object implementing the `dataset_protocol` protocol and returns a classifier term that can be used for prediction and exported as predicate clauses.

Availability:
`logtalk_load(linear_svm_classifier(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *classifier_common*

Uses:
format
linear_algebra
list
numberlist
population
type

Remarks:
(none)

Inherited public predicates:
check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *logistic_regression_classifier*, *c45_classifier*, *knn_classifier*, *naive_bayes_classifier*, *nearest_centroid_classifier*, *random_forest_classifier*, *adaptive_boosting_classifier*

1.205 linter_reporter

object

1.205.1 `linter_reporter`

Intercepts compiler linter warnings and caches them as machine-readable diagnostics.

Availability:
`logtalk_load(linter_reporter(loader))`

Author: Paulo Moura
Version: 1:0:3
Date: 2026-08-12

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *tool_diagnostics_common*
`public` *tutor_explanations*
`public` *options*

Provides:
logtalk::message_hook/4

Uses:
list
logtalk
os
term_io
type
user

Remarks:

- Usage: Load this tool before compiling code to be checked by the built-in linter. Call `enable/0-1` before compiling code, `disable/0` when finished collecting warnings, and then query the cached warnings using either the legacy warning predicates or the diagnostics protocol predicates. The standalone `sarif` tool can generate SARIF reports by querying these diagnostics.
- Diagnostics targets: The diagnostics predicates accept the targets `all`, `entity(Entity)`, `file(File)`, `directory(Directory)`, `rdirectory(Directory)`, `library(Library)`, and `rlibrary(Library)`. These targets simply filter the cached diagnostics collected in the current warning collection session.

Inherited public predicates:

check_option/1 check_options/1 default_option/1 default_options/1 diagnostic/2 diagnostic/3 diagnostic_rule/5 diagnostic_rules/1 diagnostic_target/1 diagnostics/2 diagnostics/3 diagnostics_preflight/2 diagnostics_preflight/3 diagnostics_summary/2 diagnostics_summary/3 diagnostics_tool/5 explain//1 option/2 option/3 valid_option/1 valid_options/1

- *Public predicates*
  - *enable/0*
  - *enable/1*
  - *disable/0*
  - *reset/0*
  - *warning/1*
  - *warnings/1*
  - *summary/1*
- *Protected predicates*
- *Private predicates*
  - *enabled_/0*
  - *warning_sequence_/1*
  - *recorded_warning_/4*
  - *collection_options_/1*
- *Operators*

Public predicates

enable/0

Enables warning collection and starts a fresh warning collection session using the default options.

Compilation flags:

`static`

Mode and number of proofs:

`enable` - `one`

enable/1

Enables warning collection and starts a fresh warning collection session using the given options.

Compilation flags:
`static`

Template:
`enable(Options)`
Mode and number of proofs:
`enable(+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is not a valid option:
`domain_error(option,Option)`

---

disable/0

Disables warning collection while preserving the cached warnings for later querying.

Compilation flags:
`static`

Mode and number of proofs:
`disable` - `one`

---

reset/0

Clears all cached warnings collected in the current session.

Compilation flags:
`static`

Mode and number of proofs:
`reset` - `one`

---

warning/1

Enumerates normalized linter warnings collected in the current session.

Compilation flags:
`static`

Template:
`warning(Warning)`
Mode and number of proofs:
`warning(-compound)` - `zero_or_more`

---

warnings/1

Returns the collected normalized linter warnings.

Compilation flags:
`static`

Template:
`warnings(Warnings)`
Mode and number of proofs:
`warnings(-list(compound))` - `one`

---

summary/1

Returns a machine-readable summary for the collected linter warnings.

Compilation flags:
`static`

Template:
`summary(Summary)`
Mode and number of proofs:
`summary(-compound)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

enabled_/0

True when warning collection is enabled.

Compilation flags:
`dynamic`

Mode and number of proofs:
`enabled_` - `zero_or_one`

---

warning_sequence_/1

Stores the last assigned warning sequence number.

Compilation flags:
`dynamic`

Template:
`warning_sequence_(Sequence)`
Mode and number of proofs:
`warning_sequence_(-integer)` - `zero_or_one`

---

recorded_warning_/4

Caches collected warnings together with their sequence number, flag, normalized message term, and printed message tokens.

Compilation flags:
`dynamic`

Template:
`recorded_warning_(Sequence,Flag,Message,Tokens)`
Mode and number of proofs:
`recorded_warning_(?integer,?atom,?compound,?list(compound))` - `zero_or_more`

---

collection_options_/1

Stores the merged options for the current warning collection session.

Compilation flags:
`dynamic`

Template:
`collection_options_(Options)`
Mode and number of proofs:
`collection_options_(-list(compound))` - `zero_or_one`

Operators

(none)

1.206 listing

category

1.206.1 listing

Listing predicates.

Availability:
`logtalk_load(listing(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2024-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *listing/0*

    - *listing/1*
    - *portray_clause/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

listing/0

Lists all clauses of all visible dynamic predicates to the current output stream.

Compilation flags:
`static`

Mode and number of proofs:
`listing` - `one`

---

listing/1

Lists all clauses of a visible dynamic predicate or non-terminal to the current output stream. When the argument is a clause head, lists all matching clauses.

Compilation flags:
`static`

Template:
`listing(Spec)`

Mode and number of proofs:
`listing(+predicate_indicator)` - `one_or_error`
`listing(+non_terminal_indicator)` - `one_or_error`
`listing(+callable)` - `one_or_error`

Exceptions:
`Spec` is not a ground term:
`instantiation_error`
`Spec` is ground but not a valid predicate indicator:
`type_error(predicate_indicator,Spec)`
`Spec` is ground but not a valid non-terminal indicator:
`type_error(non_terminal_indicator,Spec)`
`Spec` is a predicate indicator but not a visible predicate:
`existence_error(predicate,Spec)`
`Spec` is a non-terminal indicator but not a visible non-terminal:
`existence_error(non_terminal,Spec)`
`Spec` is a callable term with a `Functor/Arity` indicator but not a visible predicate:
`existence_error(predicate,Functor/Arity)`

`Spec` is a predicate indicator of a visible predicate but not a dynamic predicate:
`permission_error(access,predicate,Spec)`
`Spec` is a non-terminal indicator of a visible non-terminal but not a dynamic non-terminal:
`permission_error(access,non_terminal,Spec)`
`Spec` is a callable term for a visible predicate with a `Functor/Arity` indicator but not a dynamic predicate:
`permission_error(access,predicate,Functor/Arity)`

`portray_clause/1`

Pretty prints a clause to the current output stream.

Compilation flags:
`static`

Template:
`portray_clause(Clause)`
Mode and number of proofs:
`portray_clause(+clause)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.207 local_optimization

object

1.207.1 `barzilai_borwein(Problem)`

- `Problem` - Problem object implementing `local_optimization_problem_protocol` and defining `gradient/2`.

Barzilai-Borwein (BB) gradient method with adaptive step sizes. Requires the problem to define `gradient/2`. Supports box constraints via projection, minimization and maximization, and BB1 / BB2 / alternating formulas.

Availability:
`logtalk_load(local_optimization(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *local_optimization_solver(Problem)*

Uses:
linear_algebra
list

Remarks:

- Step formulas: `formula(bb1)` uses `alpha = (s*s)/(s*y)` (long step). `formula(bb2)` uses `alpha = (s*y)/(y*y)` (short step). `formula(alternate)` (default) switches between BB1 and BB2 each iteration.
- Safeguards: When the denominator is near zero or the computed step is outside `[step_min, step_max]`, the previous accepted step (or `step_size` on the first iteration) is reused.
- Line search: Optional non-monotone Armijo acceptance controlled by `line_search(none|armijo)`. The default `none` accepts the pure BB step.
- Bounds: When the problem defines `position_bounds/1`, trial points are projected onto the box after each step.

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

local_optimization_problem_protocol, *local_optimization_solver(Problem)*, *gradient_descent(Problem)*, *conjugate_gradient(Problem)*, *nelder_mead(Problem)*

object

1.207.2 `bfgs(Problem)`

- `Problem` - Problem object implementing `local_optimization_problem_protocol` and defining `gradient/2`.

BFGS (Broyden-Fletcher-Goldfarb-Shanno) dense quasi-Newton local optimizer with backtracking Armijo line search. Requires the problem to define `gradient/2`. Supports optional box constraints via projection, minimization and maximization.

Availability:

```
logtalk_load(local_optimization(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *local_optimization_solver(Problem)*

Uses:

linear_algebra
list

Remarks:

- Update: Maintains a dense approximation to the inverse Hessian, updated after every accepted step with the standard BFGS rank-two formula. The approximation starts at the identity matrix.
- Internal minimization form: Maximization is handled by internally minimizing the negated objective and gradient; the quasi-Newton direction, curvature test, and Armijo condition are therefore always expressed in minimization form, which avoids sign errors in the line search.
- Curvature safeguard: Whenever the curvature condition `y . s > 0` is not comfortably satisfied (possible here since the line search only enforces sufficient decrease, not a Wolfe curvature condition), the inverse-Hessian approximation is reset to the identity matrix and the next step falls back to steepest descent, rather than keeping a stale approximation that would otherwise keep producing the same near-zero-progress direction.
- Restarts: The `restart(N)` option (off by default) periodically resets the inverse-Hessian approximation to the identity matrix, exactly as `conjugate_gradient(_)` resets its search direction.
- Bounds: When the problem defines `position_bounds/1`, trial points are projected onto the box after each step. Projection can weaken the quasi-Newton model; a pure bound-constrained formulation (L-BFGS-B style) is not implemented.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

local_optimization_problem_protocol, *local_optimization_solver(Problem)*, *gradient_descent(Problem)*, *conjugate_gradient(Problem)*

object

1.207.3 `conjugate_gradient(Problem)`

- `Problem` - Problem object implementing `local_optimization_problem_protocol` and defining `gradient/2`.

Nonlinear conjugate-gradient local optimizer (Fletcher-Reeves and Polak-Ribière). Requires the problem to define `gradient/2`. Supports optional box constraints via projection, minimization and maximization, and periodic or automatic restarts.

Availability:
`logtalk_load(local_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *local_optimization_solver(Problem)*

Uses:
linear_algebra
list

Remarks:

- Beta formulas: The `beta(fletcher_reeves)` and `beta(polak_ribiere)` options select the conjugacy coefficient. Polak-Ribière uses the standard non-negative truncation `max(Beta, 0)`.
- Restarts: The search direction is reset to steepest descent every `restart(N)` iterations (default: dimension) and whenever the new direction is insufficiently downhill.
- Line search: Backtracking Armijo line search (same parameters as gradient descent).
- Bounds: When the problem defines `position_bounds/1`, trial points are projected onto the box after each step.

Inherited public predicates:

check_option/1 check_options/1 default_option/1 default_options/1 option/2 option/3 run/2 run/3 run/4 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

local_optimization_problem_protocol, *local_optimization_solver(Problem)*, *bfgs(Problem)*, *gradient_descent(Problem)*, *nelder_mead(Problem)*

object

1.207.4 `gradient_descent(Problem)`

- `Problem` - Problem object implementing `local_optimization_problem_protocol` and defining `gradient/2`.

Gradient descent (steepest descent) local optimizer with optional backtracking Armijo line search. Requires the problem to define `gradient/2`. Supports box constraints via projection, minimization and maximization.

Availability:

```
logtalk_load(local_optimization(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *local_optimization_solver(Problem)*

Uses:
linear_algebra
list

Remarks:

- Line search: The `line_search(armijo)` option (default) uses backtracking Armijo line search. The `line_search(fixed)` option uses a constant step size given by `step_size(S)`.
- Bounds: When the problem defines `position_bounds/1`, the gradient step is projected onto the box (projected gradient descent).
- Direction: For minimization the search direction is `-gradient`; for maximization it is `+gradient`.

Inherited public predicates:
check_option/1 check_options/1 default_option/1 default_options/1 option/2 option/3 run/2 run/3 run/4 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

local_optimization_problem_protocol, *local_optimization_solver(Problem)*, *bfgs(Problem)*, *conjugate_gradient(Problem)*, *nelder_mead(Problem)*

object

1.207.5 `lbfgs(Problem)`

- `Problem` - Problem object implementing `local_optimization_problem_protocol` and defining `gradient/2`.

L-BFGS (limited-memory Broyden-Fletcher-Goldfarb-Shanno) quasi-Newton local optimizer with backtracking Armijo line search. Requires the problem to define `gradient/2`. Supports optional box constraints via projection, minimization and maximization.

Availability:
`logtalk_load(local_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *local_optimization_solver(Problem)*
Uses:
linear_algebra
list

Remarks:

- Update: Instead of maintaining a dense inverse-Hessian approximation like `bfgs(_)`, only the last `memory_size(M)` step/gradient-difference pairs `(s, y)` are kept, and the search direction is recovered from them with the standard two-loop recursion (Nocedal and Wright, Algorithm 7.4). Memory and per-iteration cost are `O(M*n)` instead of `bfgs(_)`'s `O(n^2)`.
- Internal minimization form: Maximization is handled by internally minimizing the negated objective and gradient, so the two-loop recursion, curvature test, and Armijo condition are always expressed in minimization form, which avoids sign errors in the line search.
- Curvature safeguard: Whenever the curvature condition `y . s > 0` is not comfortably satisfied (possible here since the line search only enforces sufficient decrease, not a Wolfe curvature condition), the pair history is cleared and the next step falls back to steepest descent, rather than keeping a stale history that would otherwise keep producing the same near-zero-progress direction.
- Restarts: The `restart(N)` option (off by default) periodically clears the pair history, exactly as `bfgs(_)` resets its inverse-Hessian approximation to the identity.
- Bounds: When the problem defines `position_bounds/1`, trial points are projected onto the box after each step. Projection can weaken the quasi-Newton model; a pure bound-constrained formulation (L-BFGS-B style) is not implemented.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

local_optimization_problem_protocol, *local_optimization_solver(Problem)*, *gradient_descent(Problem)*, *conjugate_gradient(Problem)*, *bfgs(Problem)*

object

1.207.6 `lbfgs_b(Problem)`

- `Problem` - Problem object implementing `local_optimization_problem_protocol` with `gradient/2` (and `position_bounds/1` when box constraints are present).

L-BFGS-B bound-constrained limited-memory quasi-Newton optimizer with a level-B approximate generalized Cauchy point (first-segment quadratic min along the projected gradient path), free-set identification at the Cauchy point, feasible-step limiting, and L-BFGS two-loop recursion. Requires `gradient/2`.

Availability:

```
logtalk_load(local_optimization(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *local_optimization_solver(Problem)*

Uses:

linear_algebra
list

Remarks:

- Algorithm: Each iteration (1) builds an approximate generalized Cauchy point by minimizing a quadratic model of the limited-memory BFGS Hessian along the first segment of the projected steepest-descent path, (2) identifies the free set at that point, (3) computes an L-BFGS direction via the two-loop recursion and zeroes components outside the free set or that would leave the box, (4) uses the largest feasible step as the Armijo upper bound, and (5) updates the pair history only when the curvature condition holds.
- Approximate GCP: Level-B approximation: breakpoints along `x(t) = P(x - t g)`, univariate quadratic minimization on `[0, t1]` (first breakpoint), with curvature `d'Bd` estimated from the most recent L-BFGS pair (`gamma`). Full multi-segment BLNZ Cauchy search is not implemented.
- Versus `lbfgs(_)`: Plain `lbfgs(_)` only clamps trial points after an unconstrained step. This solver never proposes an infeasible step, stops on the projected gradient norm, and chooses the free set from an approximate Cauchy point.
- Internal minimization form: Maximization is handled by minimizing the negated objective and gradient (phi-space), as in `bfgs(_)` and `lbfgs(_)`.
- Unbounded problems: When the problem does not define `position_bounds/1`, the solver behaves like unconstrained `lbfgs(_)` (no GCP / free-set masking). Prefer `lbfgs(_)` for purely unconstrained work; prefer this solver when box constraints are present.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

local_optimization_problem_protocol, *local_optimization_solver(Problem)*, *lbfgs(Problem)*, *bfgs(Problem)*, *gradient_descent(Problem)*

protocol

1.207.7 `local_optimization_problem_protocol`

Protocol for continuous local-optimization problem definitions. A problem object must define the required predicates and may optionally define gradient, Hessian, bounds, stopping, and progress predicates. The same problem object can be used by any solver in the local_optimization library and by the existing global metaheuristics (PSO, DE, …).

Availability:
`logtalk_load(local_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-17

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *initial_point/1*
  - *objective/2*
  - *gradient/2*
  - *hessian/2*
  - *position_bounds/1*
  - *stop_condition/3*
  - *progress/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

initial_point/1

Returns a starting point for the local search. The point must be a non-empty list of numbers. When position bounds are defined, the point must lie inside those bounds.

Compilation flags:

```
static
```

Template:

```
initial_point(Point)
```

Mode and number of proofs:

```
initial_point(-list(number)) - one
```

---

objective/2

Computes the objective (cost) value of a point. Solvers minimize this value by default; use the `objective(maximize)` option to maximize it instead. The returned value must be a number.

Compilation flags:

```
static
```

Template:

```
objective(Point,Value)
```

Mode and number of proofs:

```
objective(+list(number),-number) - one
```

---

gradient/2

Optional. Computes the gradient of the objective at a point. Required by gradient-based solvers (gradient descent, conjugate gradient, BFGS, …). When not defined those solvers raise an existence error. The returned vector must have the same length as the point.

Compilation flags:

```
static
```

Template:

```
gradient(Point,Gradient)
```

Mode and number of proofs:

```
gradient(+list(number),-list(number)) - zero_or_one
```

---

hessian/2

Optional for most solvers. Computes the Hessian matrix of the objective at a point (list of rows). Required by trust-region Newton-CG; when not defined that solver raises an existence error. Not used by any other solver.

Compilation flags:

```
static
```

Template:

```
hessian(Point,Hessian)
```

Mode and number of proofs:

```
hessian(+list(number),-list(list(number))) - zero_or_one
```

---

position_bounds/1

Optional. Returns one `Lower-Upper` numeric bound pair per dimension, with `Lower =< Upper`. When defined, solvers that support bounds project or clamp trial points onto the box. When not defined the search is treated as unbounded.

Compilation flags:

```
static
```

Template:

```
position_bounds(Bounds)
```

Mode and number of proofs:

```
position_bounds(-list(pair)) - zero_or_one
```

---

stop_condition/3

Optional. True when the search should stop given the completed iteration count, the current best point, and its objective value. When not defined the solver runs until its own termination criteria (maximum iterations, tolerances, …) are met.

Compilation flags:

```
static
```

Template:

```
stop_condition(Iteration,BestPoint,BestValue)
```

Mode and number of proofs:

```
stop_condition(+non_negative_integer,+list(number),+number) - zero_or_one
```

`progress/5`

Optional. Called periodically to report progress. Arguments are the completed iteration count, current best point, best objective value, a solver-specific measure of step size or gradient norm, and the number of objective evaluations so far. A final call is made when the solver terminates if progress reporting is enabled.

Compilation flags:
`static`

Template:
`progress(Iteration,BestPoint,BestValue,Measure,Evaluations)`

Mode and number of proofs:
`progress(+non_negative_integer,+list(number),+number,+number,+number)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

local_optimization_solver(Problem), *barzilai_borwein(Problem)*, *bfgs(Problem)*, *conjugate_gradient(Problem)*, *gradient_descent(Problem)*, *lbfgs(Problem)*, *lbfgs_b(Problem)*, *nelder_mead(Problem)*, *trust_region_newton_cg(Problem)*

category

1.207.8 `local_optimization_solver(Problem)`

- `Problem` - Problem object implementing `local_optimization_problem_protocol`.

Common code for local-optimization solvers: option handling, objective-direction helpers, bound projection, vector utilities, progress reporting, and basic validation. Concrete solvers import this category and implement the search loop.

Availability:
`logtalk_load(local_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-24

Compilation flags:
`static`

Extends:
`public` *options*
Uses:
list
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *run/2*
  - *run/3*
  - *run/4*
- *Protected predicates*
  - *initial_point/2*
  - *better_value/3*
  - *target_reached/3*
  - *project_to_bounds/3*
  - *check_bounds/1*
  - *check_point/2*
  - *report_progress/6*
  - *report_final/5*
  - *progress_hook/5*
- *Private predicates*
- *Operators*

Public predicates

run/2

Runs the solver with default options and returns the best point and objective value found.

Compilation flags:
`static`

Template:
`run(BestPoint,BestValue)`
Mode and number of proofs:
`run(-list(number),-number)` - `one`

run/3

Runs the solver with the given options and returns the best point and objective value found.

Compilation flags:
`static`

Template:
`run(BestPoint,BestValue,Options)`
Mode and number of proofs:
`run(-list(number),-number,+list(compound))` - `one`

---

run/4

Runs the solver with the given options and returns the best point, objective value, and run statistics.

Compilation flags:
`static`

Template:
`run(BestPoint,BestValue,Statistics,Options)`
Mode and number of proofs:
`run(-list(number),-number,-list(compound),+list(compound))` - `one`

Protected predicates

initial_point/2

Returns the initial point defined by the `initial_point/1` option if present. Otherwise returns the problem defined initial point.

Compilation flags:
`static`

Template:
`initial_point(Options,Point)`
Mode and number of proofs:
`initial_point(+list(compound),-list(number))` - `one`

---

better_value/3

True when `Value` is strictly better than `Reference` according to the optimization direction.

Compilation flags:
`static`

Template:
`better_value(Objective,Value,Reference)`
Mode and number of proofs:
`better_value(+atom,+number,+number)` - `zero_or_one`

---

target_reached/3

True when the current value has reached or passed the optional target in the selected direction.

Compilation flags:
`static`

Template:
`target_reached(Objective,Value,Target)`
Mode and number of proofs:
`target_reached(+atom,+number,+term)` - `zero_or_one`

---

project_to_bounds/3

Clamps each component of a point to the corresponding `Low-High` interval. When `Bounds` is empty the point is returned unchanged.

Compilation flags:
`static`

Template:
`project_to_bounds(Point,Bounds,Projected)`
Mode and number of proofs:
`project_to_bounds(+list(number),+list(pair),-list(number))` - `one`

---

check_bounds/1

Checks that the bounds are valid.

Compilation flags:
`static`

Template:
`check_bounds(Bounds)`
Mode and number of proofs:
`check_bounds(+list(pair))` - `one_or_error`

Exceptions:
`Bounds` is a variable:
`instantiation_error`
`Bounds` is neither a variable nor a valid list of bounds:
`domain_error(position_bounds,Bounds)`

---

check_point/2

Checks that the bounds are valid. Assumes `Point` is already checked to be a list of numbers.

Compilation flags:
`static`

Template:
`check_point(Point,Bounds)`
Mode and number of proofs:
`check_point(+list(number),+list(pair))` - `one_or_error`

Exceptions:
`Point` is not a valid point:
`domain_error(initial_point,Point)`

---

report_progress/6

Reports solver progress by calling `progress_hook/5` if defined.

Compilation flags:
`static`

Template:
`report_progress(Iteration,UpdateInterval,BestPoint,BestValue,Measure,Evaluations)`
Mode and number of proofs:

```
report_progress(+non_negative_integer,+non_negative_integer,+list(number),
+number,+number,+non_negative_integer) - one
```

---

report_final/5

Reports the final solver solution by calling `progress_hook/5` if defined.

Compilation flags:

```
static
```

Template:

```
report_final(Iteration,UpdateInterval,BestPoint,BestValue,Measure)
```

Mode and number of proofs:

```
report_final(+non_negative_integer,+non_negative_integer,+list(number),+number,
+number) - one
```

---

progress_hook/5

User-defined hook predicate to report a solver step.

Compilation flags:

```
static
```

Template:

```
progress_hook(Iteration,BestPoint,BestValue,Measure,Evaluations)
```

Mode and number of proofs:

```
progress_hook(+non_negative_integer,+list(number),+number,+number,
+non_negative_integer) - zero_or_one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

local_optimization_problem_protocol

object

1.207.9 nelder_mead(Problem)

- Problem - Problem object implementing local_optimization_problem_protocol.

Nelder-Mead (downhill simplex) derivative-free local optimizer for continuous problems. Supports optional box constraints via projection, minimization and maximization, and the standard reflection / expansion / contraction / shrink operators.

Availability:

logtalk_load(local_optimization(loader))

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-03

Compilation flags:

static, context_switching_calls

Imports:

public *local_optimization_solver(Problem)*

Uses:

linear_algebra
list
pairs

Remarks:

- Algorithm: Classic Nelder-Mead simplex method. An optional adaptive-parameter variant (Gao-Han) can be selected with the adaptive(true) option.
- Bounds: When the problem defines position_bounds/1, trial points are projected onto the box after every operator.
- Initial simplex: Built from the problem initial_point/1 by stepping along each coordinate axis. The step size is controlled by the initial_step(S) option (default 0.05 of each finite range, or an absolute 0.05 when unbounded).

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

local_optimization_problem_protocol, *local_optimization_solver(Problem)*, *gradient_descent(Problem)*

object

1.207.10 `trust_region_newton_cg(Problem)`

- `Problem` - Problem object implementing `local_optimization_problem_protocol` and defining `gradient/2` and `hessian/2`.

Trust-region Newton-CG local optimizer (Steihaug-CG for the subproblem). Requires the problem to define `gradient/2` and `hessian/2`. Supports optional box constraints via projection, minimization and maximization.

Availability:
`logtalk_load(local_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-24

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *local_optimization_solver(Problem)*

Uses:
linear_algebra
list

Remarks:

- Subproblem: At each outer iteration, the step is obtained by approximately minimizing the local quadratic model within a ball of radius `trust_radius`, using the Steihaug-CG method (Nocedal and Wright, Algorithm 7.2): plain conjugate gradient on the model, terminated early either by a negative-curvature direction or by reaching the trust-region boundary, in which case the step is extended to the boundary along the current CG direction.
- No line search: Unlike the other gradient-based solvers in this library, this solver never backtracks a step size; the trust-region radius itself is grown or shrunk each iteration based on how well the quadratic model predicted the actual objective change, and a step is accepted only when that agreement is good enough.

- Internal minimization form: Maximization is handled by internally minimizing the negated objective, gradient, and Hessian, so the subproblem and acceptance test are always expressed in minimization form, which avoids sign errors.
- Convergence: Because it uses exact second-order information, this solver typically converges in far fewer iterations than `gradient_descent(_)`, `bfgs(_)`, or `lbfgs(_)` on well-behaved problems, at the cost of requiring an explicit `hessian/2`.
- Bounds: When the problem defines `position_bounds/1`, trial points are projected onto the box after each step. Projection can weaken the trust-region model agreement (the accepted step may differ from the one the subproblem solved for), which can trigger more radius shrinkage than an unconstrained problem would; a pure bound-constrained formulation is not implemented.

Inherited public predicates:

check_option/1 check_options/1 default_option/1 default_options/1 option/2 option/3 run/2 run/3 run/4 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

local_optimization_problem_protocol, *local_optimization_solver(Problem)*, *gradient_descent(Problem)*, *conjugate_gradient(Problem)*, *bfgs(Problem)*, *lbfgs(Problem)*

1.208 lof_anomaly_detector

object

1.208.1 lof_anomaly_detector

Local Outlier Factor anomaly detector with multiple distance metrics, mixed-feature support, and missing-value handling. Learns from baseline training examples selected from a dataset object implementing the `anomaly_dataset_protocol` protocol and returns a detector term that can be used for scoring, prediction, and export.

Availability:
`logtalk_load(lof_anomaly_detector(loader))`

Author: Paulo Moura
Version: 2:0:1
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *anomaly_detector_common*

Uses:
avltree
binary_heap_max
format
list
numberlist
pairs
type

Remarks:
(none)

Inherited public predicates:
anomaly_detector_options/2 *check_anomaly_detector/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict/4* *print_anomaly_detector/1* *score/3* *score_all/3* *valid_anomaly_detector/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *extract_values/3*
  - *check_feature_values/2*
  - *check_instance_ids/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

extract_values/3

Extracts feature values in the declared attribute order.

Compilation flags:
`static`

Template:
`extract_values(Attributes,AttributeValues,Values)`
Mode and number of proofs:
`extract_values(+list(atom),+list(pair),-list)` - `one_or_error`

Exceptions:
A declared attribute is absent from `AttributeValues`:
`existence_error(attribute,Attribute)`

---

check_feature_values/2

Checks prepared numeric, categorical, and missing feature values.

Compilation flags:
`static`

Template:
`check_feature_values(FeatureTypes,Values)`
Mode and number of proofs:
`check_feature_values(+list(atom),+list)` - `one_or_error`

Exceptions:
A bound feature value in `Values` is not a ground term:
`instantiation_error`
A numeric feature value is not a number:
`type_error(number,Value)`
A categorical feature value is not an atom:
`type_error(atom,Value)`

---

check_instance_ids/2

Checks distinct integer identities for training examples.

Compilation flags:
`static`

Template:
`check_instance_ids(Instances,Seen)`
Mode and number of proofs:
`check_instance_ids(+list,+list(integer))` - `one_or_error`

Exceptions:
An example identifier in `Instances` is not a ground term:
`instantiation_error`
An example identifier is not an integer:
`type_error(integer,Id)`
An example identifier occurs more than once:
`domain_error(unique_example_id,Id)`

Operators

(none)

See also

anomaly_dataset_protocol, *anomaly_detector_protocol*, *knn_distance_anomaly_detector*, *isolation_forest_anomaly_detector*

1.209 logging

object

1.209.1 logger

Global logger object for logging events to files.

Availability:
`logtalk_load(logging(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2011-01-06

Compilation flags:
`static, context_switching_calls`

Implements:

`public` *loggingp*

Remarks:
(none)

Inherited public predicates:
define_log_file/2 *disable_logging/1* *enable_logging/1* *init_log_file/2* *log_event/2* *log_file/2* *logging/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *log_file_/2*
  - *logging_to_file_/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

log_file_/2

Table of log files.

Compilation flags:
`dynamic`

Template:
`log_file_(Alias,File)`
Mode and number of proofs:
`log_file_(?atom,?nonvar)` - `zero_or_more`

---

logging_to_file_/2

Table of logging file status for log files.

Compilation flags:
`dynamic`

Template:
`logging_to_file_(Alias,Status)`

Mode and number of proofs:
`logging_to_file_(?atom,?atom)` - `zero_or_more`

Operators

(none)
category

1.209.2 `logging`

Logging events to files category.

Availability:
`logtalk_load(logging(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2011-01-06

Compilation flags:
`static`

Implements:
`public` *loggingp*

Remarks:
(none)

Inherited public predicates:
define_log_file/2 *disable_logging/1* *enable_logging/1* *init_log_file/2* *log_event/2* *log_file/2* *logging/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *log_file_/2*
  - *logging_to_file_/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

log_file_/2

Table of log files.

Compilation flags:
`dynamic`

Template:
`log_file_(Alias,File)`
Mode and number of proofs:
`log_file_(?atom,?nonvar)` - `zero_or_more`

---

logging_to_file_/2

Table of logging file status for log files.

Compilation flags:
`dynamic`

Template:
`logging_to_file_(Alias,Status)`
Mode and number of proofs:
`logging_to_file_(?atom,?atom)` - `zero_or_more`

Operators

(none)
protocol

1.209.3 loggingp

Logging events to files protocol.

Availability:
`logtalk_load(logging(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2011-01-06

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *log_file/2*
  - *define_log_file/2*
  - *init_log_file/2*
  - *log_event/2*
  - *logging/1*
  - *enable_logging/1*
  - *disable_logging/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

log_file/2

Access to the table of log files.

Compilation flags:
`static`

Template:
`log_file(Alias,File)`
Mode and number of proofs:
`log_file(?atom,?atom)` - `zero_or_more`

define_log_file/2

Defines a log file with alias Alias and file name File. If the log file already exists, its contents are kept. Logging is enabled by default.

Compilation flags:
```
static
```

Template:
```
define_log_file(Alias,File)
```
Mode and number of proofs:
```
define_log_file(+atom,+atom) - one
```

---

init_log_file/2

Initializes a new log file with alias Alias and file name File. If the log file already exists, its contents are erased. Logging is enabled by default.

Compilation flags:
```
static
```

Template:
```
init_log_file(Alias,File)
```
Mode and number of proofs:
```
init_log_file(+atom,+atom) - one
```

---

log_event/2

Logs an event Event to a log file with alias Alias. Fails if a log file with alias Alias is not defined.

Compilation flags:
```
static
```

Template:
```
log_event(Alias,Event)
```
Mode and number of proofs:
```
log_event(+atom,+nonvar) - zero_or_one
```

---

logging/1

True if logging to file with alias Alias is enabled.

Compilation flags:
`static`

Template:
`logging(Alias)`
Mode and number of proofs:
`logging(+atom)` - `zero_or_one`

---

enable_logging/1

Enables logging to file with alias Alias. Fails if a log file with alias Alias is not defined.

Compilation flags:
`static`

Template:
`enable_logging(Alias)`
Mode and number of proofs:
`enable_logging(+atom)` - `zero_or_one`

---

disable_logging/1

Disables logging to file with alias Alias. Fails if a log file with alias Alias is not defined.

Compilation flags:
`static`

Template:
`disable_logging(Alias)`
Mode and number of proofs:
`disable_logging(+atom)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

logging

1.210 logistic_regression_classifier

object

1.210.1 `logistic_regression_classifier`

Logistic regression classifier supporting binary and multiclass classification using joint softmax training. Learns from a dataset object implementing the `dataset_protocol` protocol and returns a classifier term that can be used for prediction and exported as predicate clauses.

Availability:

```
logtalk_load(logistic_regression_classifier(loader))
```

Author: Paulo Moura
Version: 2:1:1
Date: 2026-10-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *probabilistic_classifier_common*

Uses:

format
linear_algebra
list
numberlist
pairs
population
type

Remarks:

(none)

Inherited public predicates:

check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *c45_classifier*, *knn_classifier*, *naive_bayes_classifier*, *nearest_centroid_classifier*, *random_forest_classifier*, *adaptive_boosting_classifier*

1.211 loops

object

1.211.1 `loop`

Loop control structures predicates.

Availability:
`logtalk_load(loops(loader))`

Author: Paulo Moura
Version: 1:4:1
Date: 2020-12-20

Compilation flags:
`static, context_switching_calls`

Implements:

`public` *loopp*

Remarks:
(none)

Inherited public predicates:
dowhile/2 *fordownto/3* *fordownto/4* *fordownto/5* *foreach/3* *foreach/4* *forto/3* *forto/4* *forto/5* *whiledo/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.211.2 `loopp`

Loop control constructs protocol.

Availability:
`logtalk_load(loops(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2017-03-20

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *whiledo/2*
  - *dowhile/2*
  - *foreach/3*
  - *foreach/4*
  - *forto/3*
  - *forto/4*
  - *forto/5*
  - *fordownto/3*
  - *fordownto/4*
  - *fordownto/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

whiledo/2

While `Condition` is true do `Action`.

Compilation flags:
`static`

Template:
`whiledo(Condition,Action)`
Meta-predicate template:
`whiledo(0,0)`
Mode and number of proofs:
`whiledo(+callable,@callable)` - `zero_or_one`

---

dowhile/2

Do `Action` while `Condition` is true.

Compilation flags:
`static`

Template:
`dowhile(Action,Condition)`
Meta-predicate template:
`dowhile(0,0)`

Mode and number of proofs:
`dowhile(@callable,+callable)` - `zero_or_one`

---

foreach/3

For each `Element` in `List` call `Goal`.

Compilation flags:
`static`

Template:
`foreach(Element,List,Goal)`
Meta-predicate template:
`foreach(*,*,0)`
Mode and number of proofs:
`foreach(@var,+list(term),@callable)` - `zero_or_one`

---

foreach/4

For each `Element` in `List` at position `Index` call `Goal`. `Index` starts at 1.

Compilation flags:
`static`

Template:
`foreach(Element,Index,List,Goal)`
Meta-predicate template:
`foreach(*,*,*,0)`
Mode and number of proofs:
`foreach(@var,@var,+list(term),@callable)` - `zero_or_one`

---

forto/3

Calls `Goal` counting up from `First` to `Last`. Increment is 1. For convenience, `First` and `Last` can be arithmetic expressions. Fails iff `Goal` fails.

Compilation flags:
`static`

Template:
`forto(First,Last,Goal)`
Meta-predicate template:

```
forto(*,*,0)
```

Mode and number of proofs:

```
forto(+number,+number,@callable) - zero_or_one
```

---

forto/4

Calls `Goal` counting up from `First` to `Last` and binding `Count` to each successive value. Increment is 1. For convenience, `First` and `Last` can be arithmetic expressions. Fails iff `Goal` fails.

Compilation flags:

```
static
```

Template:

```
forto(Count,First,Last,Goal)
```

Meta-predicate template:

```
forto(*,*,*,0)
```

Mode and number of proofs:

```
forto(@var,+number,+number,@callable) - zero_or_one
```

---

forto/5

Calls `Goal` counting up from `First` to `Last` and binding `Count` to each successive value. For convenience, `First`, `Last`, and `Increment` can be arithmetic expressions (uses `Increment` absolute value). Fails iff `Goal` fails.

Compilation flags:

```
static
```

Template:

```
forto(Count,First,Last,Increment,Goal)
```

Meta-predicate template:

```
forto(*,*,*,*,0)
```

Mode and number of proofs:

```
forto(@var,+number,+number,+number,@callable) - zero_or_one
```

---

fordownto/3

Calls `Goal` counting down from `First` to `Last`. Decrement is 1. For convenience, `First` and `Last` can be arithmetic expressions. Fails iff `Goal` fails.

Compilation flags:

```
static
```

Template:

```
fordownto(First,Last,Goal)
```

Meta-predicate template:

```
fordownto(*,*,0)
```

Mode and number of proofs:

```
fordownto(+number,+number,@callable) - zero_or_one
```

---

fordownto/4

Calls `Goal` counting down from `First` to `Last` and binding `Count` to each successive value. Decrement is 1. For convenience, `First` and `Last` can be arithmetic expressions. Fails iff `Goal` fails.

Compilation flags:

```
static
```

Template:

```
fordownto(Count,First,Last,Goal)
```

Meta-predicate template:

```
fordownto(*,*,*,0)
```

Mode and number of proofs:

```
fordownto(@var,+number,+number,@callable) - zero_or_one
```

---

fordownto/5

Calls `Goal` counting down from `First` to `Last` and binding `Count` to each successive value. For convenience, `First`, `Last`, and `Decrement` can be arithmetic expressions (uses `Decrement` absolute value). Fails iff `Goal` fails.

Compilation flags:

```
static
```

Template:

```
fordownto(Count,First,Last,Decrement,Goal)
```

Meta-predicate template:

```
fordownto(*,*,*,*,0)
```

Mode and number of proofs:

```
fordownto(@var,+number,+number,+number,@callable) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

loop

1.212 lru_caches

object

1.212.1 `lru_cache`

Immutable LRU (Least Recently Used) caches.

Availability:
`logtalk_load(lru_caches(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *lru_cache_protocol*
Uses:
avltree
list
type

Remarks:
(none)

Inherited public predicates:
as_list/2 *capacity/2* *clear/2* *delete/4* *empty/1* *evict/3* *get/4* *lookup/3* *new/2* *put/4* *size/2* *update/5*

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

lru_cache_protocol

protocol

1.212.2 `lru_cache_protocol`

LRU (Least Recently Used) cache protocol. Key-value pairs are represented as `Key-Value`.

Availability:
`logtalk_load(lru_caches(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *new/2*
  - *capacity/2*
  - *size/2*
  - *empty/1*
  - *lookup/3*
  - *get/4*
  - *put/4*
  - *update/5*
  - *delete/4*
  - *evict/3*
  - *as_list/2*
  - *clear/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new/2

Creates a new empty cache with the given maximum capacity.

Compilation flags:

`static`

Template:

`new(Capacity,Cache)`

Mode and number of proofs:

`new(+non_negative_integer,-cache)` - `one_or_error`

Exceptions:

`Capacity` is a variable:

`instantiation_error`

`Capacity` is neither a variable nor an integer:

`type_error(integer,Capacity)`

`Capacity` is an integer but is less than zero:

`domain_error(non_negative_integer,Capacity)`

capacity/2

Returns the maximum cache capacity.

Compilation flags:
```
static
```

Template:
```
capacity(Cache,Capacity)
```
Mode and number of proofs:
```
capacity(+cache,-non_negative_integer) - one
```

---

size/2

Returns the number of entries in the cache.

Compilation flags:
```
static
```

Template:
```
size(Cache,Size)
```
Mode and number of proofs:
```
size(+cache,-non_negative_integer) - one
```

---

empty/1

True iff the cache is empty.

Compilation flags:
```
static
```

Template:
```
empty(Cache)
```
Mode and number of proofs:
```
empty(+cache) - zero_or_one
```

---

lookup/3

Looks up a matching key-value pair without changing its recency. Fails if the key is not found or the value does not unify.

Compilation flags:
`static`

Template:
`lookup(Key,Value,Cache)`
Mode and number of proofs:
`lookup(+ground,?term,+cache)` - `zero_or_one_or_error`

Exceptions:
`Key` is not a ground term:
`instantiation_error`

---

get/4

Looks up a matching key-value pair and marks it as most recently used, returning the updated cache. Fails if the key is not found or the value does not unify.

Compilation flags:
`static`

Template:
`get(Key,Value,Cache,NewCache)`
Mode and number of proofs:
`get(+ground,?term,+cache,-cache)` - `zero_or_one_or_error`

Exceptions:
`Key` is not a ground term:
`instantiation_error`

---

put/4

Inserts or replaces a key-value pair, marks it as most recently used, and evicts the least recently used entry when necessary.

Compilation flags:
`static`

Template:
`put(Key,Value,Cache,NewCache)`
Mode and number of proofs:
`put(+ground,+term,+cache,-cache)` - `one_or_error`

Exceptions:
`Key` is not a ground term:
`instantiation_error`

---

update/5

Updates an existing matching key-value pair and marks it as most recently used. Fails if the key is not found or the old value does not unify.

Compilation flags:
`static`

Template:
`update(Key,OldValue,NewValue,Cache,NewCache)`
Mode and number of proofs:
`update(+ground,?term,+term,+cache,-cache)` - `zero_or_one_or_error`

Exceptions:
`Key` is not a ground term:
`instantiation_error`

---

delete/4

Deletes a matching key-value pair, returning the updated cache. Fails if the key is not found or the value does not unify.

Compilation flags:
`static`

Template:
`delete(Key,Value,Cache,NewCache)`
Mode and number of proofs:
`delete(+ground,?term,+cache,-cache)` - `zero_or_one_or_error`

Exceptions:
`Key` is not a ground term:
`instantiation_error`

---

evict/3

Evicts and returns the least recently used key-value pair. Fails if the cache is empty.

Compilation flags:
`static`

Template:
`evict(Cache,Pair,NewCache)`
Mode and number of proofs:
`evict(+cache,-pair,-cache)` - `zero_or_one`

---

as_list/2

Returns the cache key-value pairs ordered from most recently used to least recently used.

Compilation flags:
`static`

Template:
`as_list(Cache,Pairs)`
Mode and number of proofs:
`as_list(+cache,-list(pair))` - `one`

---

clear/2

Returns an empty cache with the same capacity.

Compilation flags:
`static`

Template:
`clear(Cache,EmptyCache)`
Mode and number of proofs:
`clear(+cache,-cache)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

lru_cache

1.213 massey_ranker

object

1.213.1 `massey_ranker`

Massey pairwise preference ranker. Learns one deterministic rating per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol by solving the Massey linear system built from aggregated pairwise outcomes and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:
`logtalk_load(massey_ranker(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-08-18

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ranking_dataset_common*
`public` *score_ranker_model_common*
`public` *pairwise_strength_ranker_common*

Uses:
avltree
list
numberlist

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *colley_ranker*, *elo_ranker*

1.214 mcp_server

protocol

1.214.1 mcp_cache_protocol

Optional protocol for per-operation cache policy used by the MCP 2026-07-28 specification. An application may implement `cache_policy/4` to choose TTL (milliseconds) and scope (`public` or `private`) for complete results of `server/discover`, list operations (`tools/list`, `prompts/list`, `resources/list`, `resources/templates/list`), and `resources/read`. Cache fields are never attached to `input_required` results or MRTR retries. When the predicate is not defined, or for an operation it does not cover, the server falls back to the options `cache_ttl(0)` and `cache_scope(private)`.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-14

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- TTL: A non-negative integer number of milliseconds. `0` means the result must not be cached.
- Scope: Either `public` (may be shared across clients) or `private` (client-specific).
- Applicability: Only complete final results of discover, list, and resource-read operations receive cache fields. Tool calls, prompt gets that return `input_required`, and any MRTR continuation never include `ttlMs` / `cacheScope`.

Inherited public predicates:
(none)

- *Public predicates*
  - *cache_policy/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

cache_policy/4

Returns the cache policy for a given operation and request. `Operation` is one of `discover`, `tools_list`, `prompts_list`, `resources_list`, `resources_templates_list`, or `resources_read`. `Request` is the request-specific identifier or arguments (e.g. a resource URI for `resources_read`, or a free term for list/discover). `TTL` is a non-negative integer (milliseconds). `Scope` is `public` or `private`. Failure means the server should use its configured defaults.

Compilation flags:
`static`

Template:
`cache_policy(Operation,Request,TTL,Scope)`
Mode and number of proofs:
`cache_policy(+atom,+term,-integer,-atom)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.214.2 mcp_completion_protocol

Protocol for Logtalk objects that provide MCP prompt argument and resource URI completions. Used by the 2025-06-18, 2025-11-25, and 2026-07-28 adapters.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-02

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- Capabilities: Objects providing completions must declare `completions` in their `capabilities/1` predicate (from the `mcp_tool_protocol` protocol).
- References: The reference is either `prompt(Name)` for a declared prompt or `resource(URI)` for a declared resource URI or URI template.
- Results: The result is `completion(Values)` or `completion(Values, Total, HasMore)`. `Values` is an application-ranked list of atoms. The server preserves the first 100 values and sets `hasMore` to true when truncation is required. `Total` is a non-negative integer and `HasMore` is `true` or `false`.

Inherited public predicates:

(none)

- *Public predicates*
  - *completion/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

completion/4

Returns completion suggestions for a prompt argument or resource URI. `Reference` is `prompt(Name)` or `resource(URI)`, `Argument` is an `ArgumentName-PartialValue` pair, `Context` is a list of previously resolved `Name-Value` argument pairs, and `Result` is `completion(Values)` or `completion(Values, Total, HasMore)`.

Compilation flags:

`static`

Template:

`completion(Reference,Argument,Context,Result)`

Mode and number of proofs:

`completion(+compound,+pair,+list(pair),--compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

protocol

1.214.3 mcp_multiround_protocol

Protocol for multi-round tool/prompt/resource operations (MRTR) used by the MCP 2026-07-28 specification. Implementing objects may define the round hooks to request additional input from the client (elicitation, sampling, roots) and to resume with input responses and opaque request state. Existing applications that do not implement these hooks continue to use `tool_call/3`, `prompt_get/3`, `resource_read/3`, or auto-dispatch; the 2026 adapter wraps those results as `complete(Result)`. The 2026 adapter never invokes the 2025-06-18 `tool_call/4` elicitation API.

Availability:

`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-14

Compilation flags:

`static`

Dependencies:

(none)

Remarks:

- Request context: Each round hook receives a `request_context(ClientCapabilities, InputResponses, RequestState, Progress)` term. `ClientCapabilities` is the curly-term advertised by the client. `InputResponses` is a list of `input_response(Key, Response)` terms (or empty on the first round). `RequestState` is application-owned opaque data (or `none`). `Progress` is a callable closure for emitting progress notifications when the client supplied a progress token; it is only meaningful under the 2026 adapter.
- Complete results: A hook returns `complete(Result)` where `Result` uses the existing canonical vocabulary: `text(Atom)`, `error(Atom)`, `results(List)`, `structured(StructuredContent)`, `structured(Items, StructuredContent)` for tools; `messages(MessageList)` or `messages(Description, MessageList)` for prompts; `contents(ContentList)` for resources.
- Input-required results: A hook returns `input_required(InputRequests, RequestState)`. `InputRequests` is a nonempty list of uniquely keyed `input_request(Key, Request)` terms, and/or `RequestState` is non-`none`. An input-required result must contain a nonempty request list, non-`none` state, or both.
- Allowed request forms: `Request` may be `form_elicitation(Message, Schema)`, `url_elicitation(Message, URL)`, `sampling(Messages, ModelPreferences, SystemPrompt, IncludeContext)`, or `roots`.
- Input responses: On a subsequent round the adapter supplies `InputResponses` with the same keys. Response values are `accept(Content)`, `decline`, `cancel` (elicitation), sampling results, or roots results. Unrecognized keys are ignored. Malformed known values yield a `-32602` error. Missing required responses cause the adapter to re-issue the corresponding input requests.
- State integrity: `RequestState` is opaque to the adapter. Applications that use it for authorization or business decisions must integrity-protect, bind, and validate it themselves.

Inherited public predicates:

(none)

- *Public predicates*
  - *tool_call_round/4*
  - *prompt_get_round/4*
  - *resource_read_round/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

tool_call_round/4

Handles one round of a multi-round tool call. `Name` is the MCP tool name, `Arguments` is a list of `ArgumentName-Value` pairs, `Context` is a `request_context(ClientCapabilities, InputResponses, RequestState, Progress)` term, and `RoundResult` is unified with either `complete(Result)` or `input_required(InputRequests, RequestState)`. If not defined, the 2026 adapter falls back to `tool_call/3` (or auto-dispatch) and wraps the outcome as `complete`.

Compilation flags:

`static`

Template:

`tool_call_round(Name,Arguments,Context,RoundResult)`

Mode and number of proofs:

`tool_call_round(+atom,+list(pair),+compound,--compound)` - `zero_or_one`

---

prompt_get_round/4

Handles one round of a multi-round prompt get. `Name` is the MCP prompt name, `Arguments` is a list of `ArgumentName-Value` pairs, `Context` is a `request_context/4` term, and `RoundResult` is unified with either `complete(Result)` or `input_required(InputRequests, RequestState)`. If not defined, the 2026 adapter falls back to `prompt_get/3` and wraps the outcome as `complete`.

Compilation flags:

`static`

Template:

`prompt_get_round(Name,Arguments,Context,RoundResult)`

Mode and number of proofs:

`prompt_get_round(+atom,+list(pair),+compound,--compound)` - `zero_or_one`

---

resource_read_round/4

Handles one round of a multi-round resource read. `URI` is the resource identifier, `Arguments` is a list of `ArgumentName-Value` pairs (currently unused by most applications), `Context` is a `request_context/4` term, and `RoundResult` is unified with either `complete(Result)` or `input_required(InputRequests, RequestState)`. If not defined, the 2026 adapter falls back to `resource_read/3` and wraps the outcome as `complete`.

Compilation flags:
`static`

Template:
`resource_read_round(URI,Arguments,Context,RoundResult)`
Mode and number of proofs:
`resource_read_round(+atom,+list(pair),+compound,--compound)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.214.4 mcp_prompt_protocol

Protocol for Logtalk objects that provide prompts to be exposed via an MCP (Model Context Protocol) server. Implementing objects must define the set of prompts available and handle prompt get requests. Prompts are templates for structured LLM interactions that can accept arguments to customize their behavior. Used by both the 2025-06-18 and 2026-07-28 adapters.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-14

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- Capabilities: Objects providing prompts must declare `prompts` in their `capabilities/1` predicate (from the `mcp_tool_protocol` protocol). The server will then advertise the `prompts` capability and handle `prompts/list` and `prompts/get` requests.
- Prompt descriptors: Each prompt is described by a `prompt(Name, Description, Arguments)` or `prompt(Name, Title, Description, Arguments)` term where `Name` is an atom, `Title` is an optional human-friendly display name (an atom), `Description` is a human-readable atom, and `Arguments` is a list of `argument(ArgName, ArgDescription, Required)` terms.
- Prompt messages: The `prompt_get/3` predicate must return a result term containing a list of messages. Each message is a `message(Role, Content)` term where `Role` is `user` or `assistant` and `Content` is `text(Text)` where `Text` is an atom.
- Multi-round (2026-07-28): For the 2026-07-28 adapter, applications that need additional input during a prompt get implement `prompt_get_round/4` from the `mcp_multiround_protocol`. Existing `prompt_get/3` remains valid and is wrapped as a `complete` result when the round hook is absent.

Inherited public predicates:

(none)

- *Public predicates*
  - *prompts/1*
  - *prompt_get/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

prompts/1

Returns a list of prompt descriptors available from this object. Each descriptor is a compound term `prompt(Name, Description, Arguments)` or `prompt(Name, Title, Description, Arguments)` where `Name` is the MCP prompt name (an atom), `Title` is an optional human-friendly display name (an atom), `Description` is a human-readable description (an atom), and `Arguments` is a list of `argument(ArgName, ArgDescription, Required)` terms describing the prompt arguments. `ArgName` and `ArgDescription` are atoms, and `Required` is the boolean `true` or `false`.

Compilation flags:

`static`

Template:

`prompts(Prompts)`

Mode and number of proofs:

`prompts(-list(compound))` - `one`

prompt_get/3

Handles a prompt get request. `Name` is the MCP prompt name (as declared in `prompts/1`), `Arguments` is a list of `ArgumentName-Value` pairs provided by the client, and `Result` is unified with the prompt result. The result must be one of: `messages(MessageList)` for a list of prompt messages, or `messages(Description, MessageList)` to also include a description. Each message in the list must be a `message(Role, Content)` term where `Role` is `user` or `assistant` and `Content` is `text(Text)` where `Text` is an atom.

Compilation flags:
`static`

Template:
`prompt_get(Name,Arguments,Result)`
Mode and number of proofs:
`prompt_get(+atom,+list(pair),--compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.214.5 mcp_resource_protocol

Protocol for Logtalk objects that provide resources to be exposed via an MCP (Model Context Protocol) server. Implementing objects must define the set of resources available and handle resource read requests. Applications may also expose parameterized resources using URI templates. Resources expose data and content from the application that MCP clients can access. Used by both the 2025-06-18 and 2026-07-28 adapters.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-02

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- Capabilities: Objects providing resources must declare `resources` in their `capabilities/1` predicate (from the `mcp_tool_protocol` protocol). The server will then advertise the `resources` capability and handle `resources/list`, `resources/templates/list`, and `resources/read` requests.
- Resource descriptors: Each resource is described by a `resource(URI, Name, Description, MimeType)` or `resource(URI, Name, Title, Description, MimeType)` term where `URI` is the resource identifier (an atom), `Name` is a human-readable name (an atom), `Title` is an optional human-friendly display name (an atom), `Description` is a human-readable description (an atom), and `MimeType` is the MIME type of the resource content (an atom, e.g. `'text/plain'`).
- Resource template descriptors: Parameterized resources are optionally described by `resource_template(URITemplate, Name, Description, MimeType)` or `resource_template(URITemplate, Name, Title, Description, MimeType)` terms. `URITemplate` is an RFC 6570 URI template atom. Template listing is validated as a whole: an invalid template raises a `domain_error(uri_template, URITemplate)` error and no partial list is advertised. The server advertises valid templates without expanding them; applications validate, authorize, and read concrete expanded URIs using `resource_read/3`.
- Resource contents: The `resource_read/3` predicate must return a result term. The result must be `contents(ContentList)` where each item is `text_content(URI, MimeType, Text)` for text resources or `blob_content(URI, MimeType, Base64Data)` for binary resources encoded as base64.
- Multi-round (2026-07-28): For the 2026-07-28 adapter, applications that need additional input during a resource read implement `resource_read_round/4` from the `mcp_multiround_protocol`. Existing `resource_read/3` remains valid and is wrapped as a `complete` result when the round hook is absent.
- Caching (2026-07-28): Complete `resources/read` results may include cache fields (`ttlMs`, `cacheScope`) derived from `mcp_cache_protocol` or server options. Cache fields are never attached to `input_required` results.

Inherited public predicates:

(none)

- *Public predicates*
  - *resources/1*
  - *resource_templates/1*
  - *resource_read/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

resources/1

Returns a list of resource descriptors available from this object. Each descriptor is a compound term `resource(URI, Name, Description, MimeType)` or `resource(URI, Name, Title, Description, MimeType)` where `URI` is the resource identifier (an atom, typically a URI like `logtalk://my-app/data`), `Name` is a human-readable name (an atom), `Title` is an optional human-friendly display name (an atom), `Description` is a human-readable description (an atom), and `MimeType` is the MIME type (an atom, e.g. `'text/plain'`, `'application/json'`).

Compilation flags:

`static`

Template:

`resources(Resources)`

Mode and number of proofs:

`resources(-list(compound))` - `one`

`resource_templates/1`

Optional. Returns a list of parameterized resource descriptors. Each descriptor is a `resource_template(URITemplate, Name, Description, MimeType)` or `resource_template(URITemplate, Name, Title, Description, MimeType)` term. `URITemplate` is an RFC 6570 URI template atom. Concrete expanded URIs are handled by `resource_read/3`.

Compilation flags:

`static`

Template:

`resource_templates(ResourceTemplates)`

Mode and number of proofs:

`resource_templates(-list(compound))` - `zero_or_one`

`resource_read/3`

Handles a resource read request. `URI` is the resource identifier (as declared in `resources/1`), `Arguments` is a list of `ArgumentName-Value` pairs (currently unused but reserved for future use), and `Result` is unified with the resource result. The result must be `contents(ContentList)` where each content item is either `text_content(URI, MimeType, Text)` for text resources or `blob_content(URI, MimeType, Base64Data)` for binary resources encoded as base64. `Text` and `Base64Data` must be atoms.

Compilation flags:

`static`

Template:

`resource_read(URI,Arguments,Result)`

Mode and number of proofs:

`resource_read(+atom,+list(pair),--compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.214.6 `mcp_server`

MCP server facade. Selects spec and transport (stdio or Streamable HTTP).

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-09-02

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
http_server
list
type

Remarks:

- Specs: `'2025-06-18'` (default), `'2025-11-25'`, and `'2026-07-28'` via `spec/1`.
- Transports: `stdio` (default) and `streamable_http` via `transport/1`.
- OAuth: Streamable HTTP servers can be protected using the `oauth/4` option.

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *start/2*
  - *start/3*
  - *start/4*
  - *start/5*
  - *notify/1*
- *Protected predicates*
- *Private predicates*
  - *active_transport_object_/1*
- *Operators*

Public predicates

start/2

Starts the MCP server with default options.

Compilation flags:
`static`

Template:
`start(Name,Application)`
Mode and number of proofs:
`start(+atom,+object_identifier)` - `one`

---

start/3

Starts the MCP server with options (`spec/1`, `transport/1`, HTTP options, …).

Compilation flags:
`static`

Template:
`start(Name,Application,Options)`
Mode and number of proofs:
`start(+atom,+object_identifier,+list)` - `one_or_error`

Exceptions:
The `Options` list specifies an invalid MCP server configuration:
`domain_error(mcp_server_configuration,Version-Transport)`

---

start/4

Starts with custom streams and default options.

Compilation flags:
`static`

Template:
`start(Name,Application,Input,Output)`
Mode and number of proofs:
`start(+atom,+object_identifier,+stream,+stream)` - `one`

---

start/5

Starts with custom streams and options.

Compilation flags:
`static`

Template:
`start(Name,Application,Input,Output,Options)`
Mode and number of proofs:
`start(+atom,+object_identifier,+stream,+stream,+list)` - `one_or_error`

Exceptions:
The `Options` list specifies a unsupported MCP server configuration:
`domain_error(mcp_server_configuration,Version-Transport)`

---

notify/1

Publishes an application event to the active adapter.

Compilation flags:
`static`

Template:
`notify(Event)`
Mode and number of proofs:
`notify(+compound)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

active_transport_object_/1

Currently active transport object.

Compilation flags:
`dynamic`

Template:
`active_transport_object_(TransportObject)`
Mode and number of proofs:
`active_transport_object_(?object_identifier)` - `zero_or_one`

Operators

(none)
object

1.214.7 mcp_server_2025_06_18_spec

MCP 2025-06-18 protocol handler. Transport-agnostic message handling; returns abstract outcomes for stdio or Streamable HTTP transports to render. Synchronous elicitation requires `stdio_input/1` and `stdio_output/1` options.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *mcp_server_spec_protocol*
Imports:
`public` *mcp_server_application*
Uses:
json_rpc
list
term_io
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *cleanup/0* *default_option/1* *default_options/1* *handle_message/3* *notify/1* *option/2* *option/3* *prepare/2* *spec/1* *tool_descriptors_to_json/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *supported_specs/1*
  - *current_options/1*
  - *elicit_request/5*
- *Protected predicates*
  - *handle_initialize/4*
  - *handle_ping/2*
  - *handle_tools_list/3*
  - *handle_tools_call/4*
  - *handle_prompts_list/3*
  - *handle_prompts_get/4*

      - *handle_completion_complete/4*
      - *handle_resources_list/3*
      - *handle_resources_templates_list/3*
      - *handle_resources_read/4*
      - *generate_elicit_id/1*
      - *build_capabilities/2*
      - *best_supported_spec/3*
    - *Private predicates*
      - *client_capabilities_/1*
      - *elicit_counter_/1*
      - *initialized_/0*
      - *server_options_/1*
    - *Operators*

Public predicates

supported_specs/1

Specs this handler can negotiate, preferred first.

Compilation flags:
`static`

Template:
`supported_specs(Versions)`
Mode and number of proofs:
`supported_specs(-list)` - `one`

---

current_options/1

Merged options established by prepare/2.

Compilation flags:
`static`

Template:
`current_options(Options)`
Mode and number of proofs:
`current_options(-list)` - `zero_or_one`

---

elicit_request/5

Sends an elicitation/create request to the client and reads the response (2025-06-18 synchronous model).

Compilation flags:
`static`

Template:
`elicit_request(Input,Output,Message,RequestedSchema,Answer)`
Mode and number of proofs:
`elicit_request(+stream,+stream,+atom,+compound,--compound)` - `one`

Protected predicates

handle_initialize/4

Handles the `initialize` request and returns a `reply/1` outcome. Overridable by later 2025 family specs (e.g. 2025-11-25).

Compilation flags:
`static`

Template:
`handle_initialize(Message,Id,Options,Outcome)`
Mode and number of proofs:
`handle_initialize(+nonvar,+nonvar,+list,-nonvar)` - `one`

---

handle_ping/2

Handles the `ping` request and returns a `reply/1` outcome.

Compilation flags:
`static`

Template:
`handle_ping(Id,Outcome)`
Mode and number of proofs:
`handle_ping(+nonvar,-nonvar)` - `one`

---

handle_tools_list/3

Handles the `tools/list` request. Overridable so later specs can enrich tool descriptors (e.g. icons).

Compilation flags:
`static`

Template:
`handle_tools_list(Id,Options,Outcome)`
Mode and number of proofs:
`handle_tools_list(+nonvar,+list,-nonvar) - one`

---

handle_tools_call/4

Handles the `tools/call` request and returns a `reply/1` outcome.

Compilation flags:
`static`

Template:
`handle_tools_call(Message,Id,Options,Outcome)`
Mode and number of proofs:
`handle_tools_call(+nonvar,+nonvar,+list,-nonvar) - one`

---

handle_prompts_list/3

Handles the `prompts/list` request. Overridable so later specs can enrich prompt descriptors (e.g. icons).

Compilation flags:
`static`

Template:
`handle_prompts_list(Id,Options,Outcome)`
Mode and number of proofs:
`handle_prompts_list(+nonvar,+list,-nonvar) - one`

---

handle_prompts_get/4

Handles the `prompts/get` request and returns a `reply/1` outcome.

Compilation flags:
`static`

Template:
`handle_prompts_get(Message,Id,Options,Outcome)`
Mode and number of proofs:
`handle_prompts_get(+nonvar,+nonvar,+list,-nonvar)` - `one`

---

handle_completion_complete/4

Handles the `completion/complete` request and returns a `reply/1` outcome.

Compilation flags:
`static`

Template:
`handle_completion_complete(Message,Id,Options,Outcome)`
Mode and number of proofs:
`handle_completion_complete(+nonvar,+nonvar,+list,-nonvar)` - `one`

---

handle_resources_list/3

Handles the `resources/list` request. Overridable so later specs can enrich resource descriptors (e.g. icons).

Compilation flags:
`static`

Template:
`handle_resources_list(Id,Options,Outcome)`
Mode and number of proofs:
`handle_resources_list(+nonvar,+list,-nonvar)` - `one`

---

handle_resources_templates_list/3

Handles the `resources/templates/list` request. Overridable so later specs can enrich resource template descriptors (e.g. icons).

Compilation flags:

`static`

Template:

`handle_resources_templates_list(Id,Options,Outcome)`

Mode and number of proofs:

`handle_resources_templates_list(+nonvar,+list,-nonvar)` - `one`

---

handle_resources_read/4

Handles the `resources/read` request and returns a `reply/1` outcome.

Compilation flags:

`static`

Template:

`handle_resources_read(Message,Id,Options,Outcome)`

Mode and number of proofs:

`handle_resources_read(+nonvar,+nonvar,+list,-nonvar)` - `one`

---

generate_elicit_id/1

Generates a new elicitation id.

Compilation flags:

`dynamic`

Template:

`generate_elicit_id(ElicitId)`

Mode and number of proofs:

`generate_elicit_id(-non_negative_integer)` - `one`

---

build_capabilities/2

Builds the server `capabilities` object from application capability atoms (tools is always present; prompts, resources, completions, and the MCP Apps UI extension are optional).

Compilation flags:
`static`

Template:
`build_capabilities(ApplicationCapabilities,Capabilities)`
Mode and number of proofs:
`build_capabilities(+list,-compound)` - `one`

---

best_supported_spec/3

Selects the highest supported spec that is less than or equal to the client-requested spec. Fails when no such spec exists.

Compilation flags:
`static`

Template:
`best_supported_spec(Supported,ClientVersion,Best)`
Mode and number of proofs:
`best_supported_spec(+list,+atom,-atom)` - `zero_or_one`

Private predicates

client_capabilities_/1

Client capabilities.

Compilation flags:
`dynamic`

Template:
`client_capabilities_(Capabilities)`
Mode and number of proofs:
`client_capabilities_(-nonvar)` - `one`

---

elicit_counter_/1

Elicitation current counter.

Compilation flags:
`dynamic`

Template:
`elicit_counter_(Counter)`
Mode and number of proofs:
`elicit_counter_(-non_negative_integer)` - `one`

---

initialized_/0

Initialization completed flag.

Compilation flags:
`dynamic`

Mode and number of proofs:
`initialized_` - `zero_or_one`

---

server_options_/1

Server options.

Compilation flags:
`dynamic`

Template:
`server_options_(Options)`
Mode and number of proofs:
`server_options_(-list(compound))` - `one`

Operators

(none)
object

1.214.8 mcp_server_2025_11_25_spec

MCP 2025-11-25 protocol handler. Extends the 2025-06-18 handler with version `2025-11-25` negotiation (still accepting `2025-06-18`), optional `serverInfo.description`, icons metadata on tools/prompts/resources (SEP-973), and URL-mode elicitation (SEP-1036). EnumSchema / ElicitResult enrichments (SEP-1330) are pass-through via application-supplied schemas. Does **not** implement sampling tool calling, experimental Tasks, or OAuth/OIDC.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-31

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *mcp_server_2025_06_18_spec*
Uses:
json_rpc

Remarks:
(none)

Inherited public predicates:
check_option/1 check_options/1 cleanup/0 current_options/1 default_option/1 default_options/1 elicit_request/5 handle_message/3 notify/1 option/2 option/3 prepare/2 spec/1 supported_specs/1 tool_descriptors_to_json/3 valid_option/1 valid_options/1

- *Public predicates*
  - *elicit_url_request/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

elicit_url_request/5

Sends a URL-mode `elicitation/create` request (SEP-1036) and reads the client response. `Answer` is `accept(Content)`, `decline`, or `cancel`.

Compilation flags:
`static`

Template:
`elicit_url_request(Input,Output,Message,URL,Answer)`
Mode and number of proofs:

```
elicit_url_request(+stream,+stream,+atom,+atom,--compound) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.214.9 mcp_server_2026_07_28_spec

MCP 2026-07-28 protocol handler. Returns reply/1, reply_with_progress/2, subscribe/3, accepted, or no_reply outcomes. Does not write to streams; transports render outcomes.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *mcp_server_spec_protocol*
Imports:
`public` *mcp_server_application*
Uses:
json_rpc
list
term_io
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *cleanup/0* *default_option/1* *default_options/1* *handle_message/3* *notify/1* *option/2* *option/3* *prepare/2* *spec/1* *tool_descriptors_to_json/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *run_stdio_loop/3*
  - *emit_progress/6*
- *Protected predicates*
- *Private predicates*
  - *output_stream_/1*
  - *subscription_/3*
  - *running_/0*
  - *reply_outcome_/1*
  - *progress_buffer_/1*
  - *current_options_/1*
- *Operators*

Public predicates

run_stdio_loop/3

Stdio read/dispatch loop. Called by the stdio transport.

Compilation flags:
```
static
```

Template:
```
run_stdio_loop(Input,Output,Options)
```
Mode and number of proofs:
```
run_stdio_loop(+stream,+stream,+list) - one
```

---

emit_progress/6

Buffers a notifications/progress event or invokes progress_hook/1 from current options.

Compilation flags:
```
static
```

Template:
```
emit_progress(Token,RequestId,Output,ProgressValue,Total,Message)
```
Mode and number of proofs:
```
emit_progress(+term,+term,+stream,+number,+number,+atom) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

output_stream_/1

Current output stream used for stdio rendering and subscription event fan-out.

Compilation flags:
`dynamic`

Template:
`output_stream_(Stream)`
Mode and number of proofs:
`output_stream_(-stream)` - `zero_or_one`

---

subscription_/3

Active subscription entry: identifier, opening request id, and filter list.

Compilation flags:
`dynamic`

Template:
`subscription_(SubscriptionId,RequestId,Filters)`
Mode and number of proofs:
`subscription_(-atom,-nonvar,-list)` - `zero_or_more`

---

running_/0

Flag set while the stdio server loop is active.

Compilation flags:
`dynamic`

Mode and number of proofs:
`running_` - `zero_or_one`

---

reply_outcome_/1

Buffered abstract outcome (`reply/1`, `subscribe/3`, etc.) for the current request.

Compilation flags:
`dynamic`

Template:
`reply_outcome_(Outcome)`
Mode and number of proofs:
`reply_outcome_(-nonvar)` - `zero_or_one`

---

progress_buffer_/1

Buffered `notifications/progress` events for the current request (before final reply).

Compilation flags:
`dynamic`

Template:
`progress_buffer_(Events)`
Mode and number of proofs:
`progress_buffer_(-list)` - `zero_or_one`

---

current_options_/1

Merged options for the active prepare/handle session.

Compilation flags:
`dynamic`

Template:
`current_options_(Options)`
Mode and number of proofs:
`current_options_(-list)` - `zero_or_one`

Operators

(none)
category

1.214.10 mcp_server_application

Common predicates for MCP servers: tool/prompt/resource descriptor conversion, completion request handling, schema derivation from `info/2` and `mode/2` directives, auto-dispatch, canonical complete-result terms, curly-term predicates, and MCP Apps (`_meta.ui`) metadata.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-07

Compilation flags:
`static`

Extends:
`public` *options*
Uses:
list
term_io
uri_template(Representation)
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *tool_descriptors_to_json/3*
- *Protected predicates*
  - *prompt_descriptors_to_json/2*
  - *resource_descriptors_to_json/3*
  - *resource_template_descriptors_to_json/2*
  - *application_resource_template_uri/2*
  - *completion_request/5*
  - *completion_result_to_json/2*
  - *auto_dispatch_tool/5*
  - *try_tool_call_3/7*
  - *tool_input_schema/4*
  - *tool_output_schema/4*

- format_content_items/2
- format_prompt_messages/2
- format_resource_contents/2
- has_pair/3
- curly_to_pairs/2
- pairs_to_curly/2

- *Private predicates*
- *Operators*

Public predicates

tool_descriptors_to_json/3

Converts a list of `tool(Name, Functor, Arity)` descriptors into MCP JSON tool definitions, deriving titles, descriptions and input schemas from the application object's `info/2` and `mode/2` directives.

Compilation flags:
`static`

Template:
`tool_descriptors_to_json(ToolDescriptors,Application,JsonTools)`
Mode and number of proofs:
`tool_descriptors_to_json(+list,+object_identifier,-list)` - `one`

Protected predicates

prompt_descriptors_to_json/2

Converts prompt descriptors (3-arg or 4-arg) into MCP JSON prompt definitions.

Compilation flags:
`static`

Template:
`prompt_descriptors_to_json(PromptDescriptors,JsonPrompts)`
Mode and number of proofs:
`prompt_descriptors_to_json(+list,-list)` - `one`

---

resource_descriptors_to_json/3

Converts resource descriptors (4-arg or 5-arg) into MCP JSON resource definitions. When the application defines `resource_ui_meta/2`, attaches `_meta.ui` (MCP Apps).

Compilation flags:

```
static
```

Template:

```
resource_descriptors_to_json(Descriptors,Application,Resources)
```

Mode and number of proofs:

```
resource_descriptors_to_json(+list,+object_identifier,-list) - one
```

---

resource_template_descriptors_to_json/2

Converts resource template descriptors (4-arg or 5-arg) into MCP JSON resource template definitions. No partial list is returned when a descriptor is invalid.

Compilation flags:

```
static
```

Template:

```
resource_template_descriptors_to_json(Descriptors,ResourceTemplates)
```

Mode and number of proofs:

```
resource_template_descriptors_to_json(+list,-list) - one_or_error
```

Exceptions:

A descriptor contains an invalid RFC 6570 URI template `URITemplate`:

```
domain_error(uri_template,URITemplate)
```

---

application_resource_template_uri/2

Checks that a concrete resource URI matches the literal segments of one of the application resource templates. Full RFC 6570 expression validation and authorization remain application responsibilities.

Compilation flags:

```
static
```

Template:

```
application_resource_template_uri(Application,URI)
```

Mode and number of proofs:

```
application_resource_template_uri(+object_identifier,+atom) - zero_or_one
```

---

completion_request/5

Validates and normalizes completion request parameters. The normalized reference is `prompt(Name)` or `resource(URI)`, the argument is a `Name-PartialValue` pair, and the context is a list of prior argument pairs.

Compilation flags:
`static`

Template:
`completion_request(Application,Params,Reference,Argument,Context)`
Mode and number of proofs:
`completion_request(+object_identifier,+compound,-compound,-pair,-list)` - `zero_or_one`

---

completion_result_to_json/2

Validates and converts a completion result to its MCP JSON curly-term representation.

Compilation flags:
`static`

Template:
`completion_result_to_json(Result,JsonCompletion)`
Mode and number of proofs:
`completion_result_to_json(+compound,-compound)` - `zero_or_one`

---

auto_dispatch_tool/5

Auto-dispatches a tool call by calling the predicate on the application, collecting output-mode arguments, and returning structured output with a backwards-compatible text content item.

Compilation flags:
`static`

Template:
`auto_dispatch_tool(Application,Functor,Arity,ToolArguments,Result)`
Mode and number of proofs:
`auto_dispatch_tool(+object_identifier,+atom,+integer,+compound,-compound)` - `one`

---

try_tool_call_3/7

Tries `tool_call/3`. Falls back to auto-dispatch on failure.

Compilation flags:
`static`

Template:
`try_tool_call_3(Application,ToolName,Functor,Arity,ArgPairs,ToolArguments,Result)`
Mode and number of proofs:
`try_tool_call_3(+object_identifier,+atom,+atom,+integer,+list,+compound,-compound)` - `one`

---

tool_input_schema/4

Derives a JSON Schema curly-term for tool input arguments from the application object's `info/2` and `mode/2` directives. Used by Streamable HTTP `x-mcp-header` / `Mcp-Param-*` validation when the application does not define `input_schema/2`.

Compilation flags:
`static`

Template:
`tool_input_schema(Application,Functor,Arity,InputSchema)`
Mode and number of proofs:
`tool_input_schema(+object_identifier,+atom,+integer,-compound)` - `one`

---

tool_output_schema/4

Derives a JSON Schema curly-term for tool output arguments from the application object's `info/2` and `mode/2` directives.

Compilation flags:
`static`

Template:
`tool_output_schema(Application,Functor,Arity,OutputSchema)`
Mode and number of proofs:
`tool_output_schema(+object_identifier,+atom,+integer,-compound)` - `one`

---

format_content_items/2

Converts canonical content item terms into MCP JSON content array elements.

Compilation flags:
```
static
```

Template:
```
format_content_items(Items,Elements)
```
Mode and number of proofs:
```
format_content_items(+list,-list) - one
```

---

format_prompt_messages/2

Converts message(Role, text(Text)) terms into MCP JSON prompt message objects.

Compilation flags:
```
static
```

Template:
```
format_prompt_messages(Messages,JsonMessages)
```
Mode and number of proofs:
```
format_prompt_messages(+list,-list) - one
```

---

format_resource_contents/2

Converts text_content/3 and blob_content/3 terms into MCP JSON resource content objects.

Compilation flags:
```
static
```

Template:
```
format_resource_contents(Contents,JsonContents)
```
Mode and number of proofs:
```
format_resource_contents(+list,-list) - one
```

---

has_pair/3

Looks up a Key-Value pair inside a curly-term.

Compilation flags:
`static`

Template:
`has_pair(Curly,Key,Value)`
Mode and number of proofs:
`has_pair(+compound,+atom,?term) - zero_or_one`

---

curly_to_pairs/2

Converts a curly-term to a list of Key-Value pairs.

Compilation flags:
`static`

Template:
`curly_to_pairs(Curly,Pairs)`
Mode and number of proofs:
`curly_to_pairs(+compound,-list) - one`

---

pairs_to_curly/2

Converts a list of Key-Value pairs to a curly-term.

Compilation flags:
`static`

Template:
`pairs_to_curly(Pairs,Curly)`
Mode and number of proofs:
`pairs_to_curly(+list,-compound) - one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.214.11 `mcp_server_spec_protocol`

Protocol for MCP server spec (aka data layer) handlers independent of transport. A handler validates and dispatches JSON-RPC messages and returns abstract outcomes that a transport (stdio or Streamable HTTP) renders to the wire.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-31

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *spec/1*
  - *prepare/2*
  - *handle_message/3*
  - *notify/1*
  - *cleanup/0*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

spec/1

MCP server spec implemented by this handler (e.g. `'2025-06-18'` or `'2026-07-28'`).

Compilation flags:
`static`

Template:
`spec(Version)`
Mode and number of proofs:
`spec(-atom)` - `one`

---

prepare/2

Initializes handler state for `Application` with merged `Options`. Does not open any transport.

Compilation flags:
`static`

Template:
`prepare(Application,Options)`
Mode and number of proofs:
`prepare(+object_identifier,+list)` - `one_or_error`

Exceptions:
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

handle_message/3

Handles one JSON-RPC Message under Options and returns an Outcome for the transport to render. Outcomes include `reply(JsonRpcTerm)`, `accepted`, `no_reply`, and transport-agnostic error replies already encoded as JSON-RPC error objects inside `reply/1`.

Compilation flags:

`static`

Template:

`handle_message(Message,Options,Outcome)`

Mode and number of proofs:

`handle_message(+nonvar,+list,-nonvar)` - `one`

---

notify/1

Publishes an application event. Transports that support server-initiated notifications deliver it; others may ignore it.

Compilation flags:

`static`

Template:

`notify(Event)`

Mode and number of proofs:

`notify(+compound)` - `zero_or_one`

---

cleanup/0

Releases handler-owned dynamic state.

Compilation flags:

`static`

Mode and number of proofs:

`cleanup` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.214.12 `mcp_server_stdio_transport`

Implements the stdio transport for MCP servers. Reads newline-delimited JSON-RPC from the input stream and writes responses to the output stream. Supports specs 2025-06-18, 2025-11-25, and 2026-07-28 selected via the `spec/1` option and delegated to `mcp_server_2025_06_18_spec`, `mcp_server_2025_11_25_spec`, or `mcp_server_2026_07_28_spec`.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-31

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *mcp_server_transport_protocol*
Uses:
json_rpc
list
mcp_server_2026_07_28_spec

Remarks:
(none)

Inherited public predicates:
cleanup/0 notify/1 spec/1 start/4

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *active_spec_object_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`active_spec_object_/1`

Spec handler object selected for the current session.

Compilation flags:
`dynamic`

Template:
`active_spec_object_(SpecObject)`
Mode and number of proofs:
`active_spec_object_(-object_identifier)` - `zero_or_one`

Operators

(none)
object

1.214.13 `mcp_server_streamable_http_transport`

Implements the Streamable HTTP transport for MCP servers. Uses Logtalk `http_server::serve_until_shutdown/5` and a dedicated `http_handler_protocol` handler object. Supports specs 2025-06-18, 2025-11-25, and 2026-07-28 selected via the `spec/1` option and delegated to the matching `mcp_server_*_spec` object. Supports optional OAuth protection and protected-resource metadata publication using the `oauth/4` option. Long-lived subscriptions/listen streams emit periodic SSE comment keep-alives (`http_sse_keepalive/1`). Requires a multi-threaded backend for subscriptions/listen.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls, threaded`

Implements:
`public` *mcp_server_transport_protocol*
Imports:
`public` *mcp_server_application*
Uses:

format
http_oauth
http_oauth_metadata
http_server
json
json_rpc
list
mcp_server_2025_06_18_spec
mcp_server_2025_11_25_spec
mcp_server_2026_07_28_spec
os
term_io
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *cleanup/0* *default_option/1* *default_options/1* *notify/1* *option/2* *option/3* *spec/1* *start/4* *tool_descriptors_to_json/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - `prepare/2`
  - `attach_sse_stream/1`
  - `detach_sse_stream/0`
  - `sse_headers/1`
  - `current_options/1`
  - `handle_mcp_request/4`
  - `emit_progress/5`
- *Protected predicates*
- *Private predicates*
  - `running_/0`
  - `server_options_/1`
  - `shutdown_control_/1`
  - `subscription_/4`
  - `progress_events_/1`
  - `progress_token_/1`
  - `sse_output_/1`
  - `sse_mode_/1`
- *Operators*

Public predicates

prepare/2

Initializes adapter state for `Application` with `Options` without opening a listener. Used by unit tests and by embeddings by pairing with `handle_mcp_request/4` and `cleanup/0`.

Compilation flags:

`static`

Template:

`prepare(Application,Options)`

Mode and number of proofs:

`prepare(+object_identifier,+list)` - `one_or_error`

Exceptions:

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

---

attach_sse_stream/1

Registers a live output stream for incremental SSE writes. Headers must already have been written. Each `emit_progress/5` call writes and flushes immediately.

Compilation flags:

`static`

Template:

`attach_sse_stream(Stream)`

Mode and number of proofs:

`attach_sse_stream(+stream)` - `one`

---

detach_sse_stream/0

Clears any live SSE output stream registration.

Compilation flags:
`static`

Mode and number of proofs:
`detach_sse_stream` - `one`

---

sse_headers/1

HTTP headers for an SSE response body.

Compilation flags:
`static`

Template:
`sse_headers(Headers)`
Mode and number of proofs:
`sse_headers(-list)` - `one`

---

current_options/1

Unified with the options list established by `prepare/2` or `start/4`.

Compilation flags:
`static`

Template:
`current_options(Options)`
Mode and number of proofs:
`current_options(-list)` - `zero_or_one`

---

handle_mcp_request/4

Handles one MCP HTTP request. `Method` is an uppercase HTTP method atom. `Headers` is a list of `Name-Value` pairs. `Body` is the raw request body atom. `HTTPResponse` is `http_response(Status, Headers, BodyAtom)` or `http_response(already_sent, Headers, BodyAtom)`.

Compilation flags:
`static`

Template:
`handle_mcp_request(Method,Headers,Body,HTTPResponse)`
Mode and number of proofs:
`handle_mcp_request(+atom,+list,+atom,-compound)` - `one`

`emit_progress/5`

Emits a notifications/progress event. In live mode the SSE record is written and flushed immediately on the attached stream. In buffered mode the event is queued until `finalize_response/4`.

Compilation flags:
`static`

Template:
`emit_progress(Token,RequestId,ProgressValue,Total,Message)`
Mode and number of proofs:
`emit_progress(+term,+term,+number,+number,+atom)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`running_/0`

True while the adapter has been prepared or is serving requests.

Compilation flags:
`dynamic`

Mode and number of proofs:
`running_` - `zero_or_one`

`server_options_/1`

Merged server options for the active prepare/start session.

Compilation flags:
`dynamic`

Template:
`server_options_(Options)`

Mode and number of proofs:
`server_options_(-list)` - `zero_or_one`

---

shutdown_control_/1

Token passed to `http_server::request_shutdown/1` when stopping the listener.

Compilation flags:
`dynamic`

Template:
`shutdown_control_(Control)`
Mode and number of proofs:
`shutdown_control_(-nonvar)` - `zero_or_one`

---

subscription_/4

Active subscriptions/listen registration. Synchronization uses `threaded_wait/1` and `threaded_notify/1` tagged by `SubscriptionId`.

Compilation flags:
`dynamic`

Template:
`subscription_(SubscriptionId,RequestId,Filters,Stream)`
Mode and number of proofs:
`subscription_(-atom,-nonvar,-list,-nonvar)` - `zero_or_more`

---

progress_events_/1

Buffered notifications/progress events when SSE mode is not live.

Compilation flags:
`dynamic`

Template:
`progress_events_(Events)`
Mode and number of proofs:
`progress_events_(-list)` - `zero_or_one`

---

progress_token_/1

progressToken from the current request, or none.

Compilation flags:
`dynamic`

Template:
`progress_token_(Token)`
Mode and number of proofs:
`progress_token_(-nonvar)` - `zero_or_one`

---

sse_output_/1

Live response stream for incremental SSE writes.

Compilation flags:
`dynamic`

Template:
`sse_output_(Stream)`
Mode and number of proofs:
`sse_output_(-stream)` - `zero_or_one`

---

sse_mode_/1

Current SSE delivery mode: live, buffered, or none.

Compilation flags:
`dynamic`

Template:
`sse_mode_(Mode)`
Mode and number of proofs:
`sse_mode_(-atom)` - `zero_or_one`

Operators

(none)

protocol

1.214.14 mcp_server_transport_protocol

Protocol for MCP server transports. The `mcp_server` facade object selects the transport via the `transport/1` option and delegates the server lifecycle to it.

Availability:

```
logtalk_load(mcp_server(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-31

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *spec/1*
  - *start/4*
  - *notify/1*
  - *cleanup/0*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

spec/1

Returns the MCP server spec version implemented by this adapter (e.g. `'2025-06-18'` or `'2026-07-28'`).

Compilation flags:

```
static
```

Template:

`spec(Version)`

Mode and number of proofs:

`spec(-atom)` - `one`

---

start/4

Starts the adapter against the given application object, input/output streams, and normalized server configuration options. Blocks until the client disconnects (EOF) or an exit condition is reached. The adapter is responsible for all request handling, capability negotiation, and cleanup of adapter-owned state.

Compilation flags:

`static`

Template:

`start(Application,Input,Output,Options)`

Mode and number of proofs:

`start(+object_identifier,+stream,+stream,+list)` - `one_or_error`

Exceptions:

The `Options` list specifies a non-supported `Version` spec:

`domain_error(protocol_version,Version)`

---

notify/1

Publishes an application event to interested clients. Canonical event terms are `tools_list_changed`, `prompts_list_changed`, `resources_list_changed`, and `resource_updated(URI)`. The 2025-06-18 adapter may ignore or reject events not supported by that specification; the 2026-07-28 adapter filters events through active subscriptions.

Compilation flags:

`static`

Template:

`notify(Event)`

Mode and number of proofs:

`notify(+compound)` - `zero_or_one`

`cleanup/0`

Releases any adapter-owned dynamic state. Called by the facade after the server loop terminates (success, failure, exception, or EOF).

Compilation flags:
`static`

Mode and number of proofs:
`cleanup` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.214.15 `mcp_streamable_http_handler`

HTTP handler bridging http_server to mcp_server_streamable_http_transport. Uses http_core response/5 with json(Term) bodies for application/json.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 0:5:0
Date: 2026-08-26

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_handler_protocol*
Uses:
format
http_core
http_process_transport
http_socket_transport
json
list
mcp_server_streamable_http_transport

term_io
user

Remarks:
(none)

Inherited public predicates:
handle/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.214.16 `mcp_tool_protocol`

Protocol for Logtalk objects that provide tools to be exposed via an MCP (Model Context Protocol) server. Implementing objects must define the set of tools available and handle tool calls. Tool metadata (names, titles, descriptions, input and output schemas) can be derived automatically from `info/2` and `mode/2` directives on the tool predicates, or overridden using `input_schema/2` and `output_schema/2`. Used by both the 2025-06-18 and 2026-07-28 adapters.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-02

Compilation flags:
`static`

Dependencies:

(none)

Remarks:

- Capabilities: Objects can optionally define `capabilities/1` to declare additional application features. Currently supported: `prompts`, `resources`, and `completions` (server capabilities advertised during initialization) and `elicitation` (a required client capability that allows the server to ask the user questions during tool execution under the 2025-06-18 adapter). If `capabilities/1` is not defined, only the `tools` server capability is advertised.
- Tool title: The tool title (human-friendly display name) is derived from a `title` key in the predicate `info/2` directive. If not specified, the predicate name (functor) is used as the title.
- Schemas: Input and output schemas are inferred from the tool predicate `info/2` and `mode/2` directives. Define `input_schema/2` or `output_schema/2` to override the inferred schema for a tool. Tools can return `structured(StructuredContent)` or `structured(ContentItems, StructuredContent)` results; the structured content is included in the `structuredContent` field of the tool call response alongside the `content` array.
- Resource links: Tool results can include `resource_link(URI, Name)` or `resource_link(URI, Name, Description, MimeType)` content items in `results/1` lists.
- Elicitation (2025-06-18 only): Under the 2025-06-18 adapter, tools that need user interaction during execution should define `tool_call/4` instead of `tool_call/3`. The extra argument is an elicitation closure that can be called as `call(Elicit, Message, Schema, Answer)` where `Message` is the prompt text (an atom), `Schema` is a JSON Schema curly-term for the requested input, and `Answer` is unified with `accept(Content)`, `decline`, or `cancel`. The 2026-07-28 adapter never invokes `tool_call/4`; multi-round interaction is provided by `mcp_multiround_protocol` instead.
- Multi-round (2026-07-28): For the 2026-07-28 adapter, applications that need additional input during a tool call implement `tool_call_round/4` from the `mcp_multiround_protocol`. Existing `tool_call/3` and auto-dispatch remain valid and are wrapped as `complete` results when the round hook is absent.

Inherited public predicates:

(none)

- *Public predicates*
  - *capabilities/1*
  - *tools/1*
  - *tool_call/3*
  - *tool_call/4*
  - *output_schema/2*
  - *input_schema/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

capabilities/1

Returns a list of additional MCP features used by this provider. `prompts`, `resources`, and `completions` are advertised as server capabilities. `elicitation` declares that the provider may ask the user questions and is enabled only when the client advertises the corresponding client capability. If not defined, only the `tools` server capability is advertised.

Compilation flags:

```
static
```

Template:

```
capabilities(Capabilities)
```

Mode and number of proofs:

```
capabilities(-list(atom)) - one
```

---

tools/1

Returns a list of tool descriptors available from this object. Each descriptor is a compound term `tool(Name, Functor, Arity)` where `Name` is the MCP tool name (an atom), `Functor` is the predicate functor, and `Arity` is the predicate arity. By default, the tool name is the predicate functor. Tool titles are derived from `info/2` `title` keys (falling back to the predicate name). Tool descriptions and parameter schemas are derived from the `info/2` and `mode/2` directives of the corresponding predicates.

Compilation flags:

```
static
```

Template:

```
tools(Tools)
```

Mode and number of proofs:

```
tools(-list(compound)) - one
```

---

tool_call/3

Handles a tool call. `Name` is the MCP tool name (as declared in `tools/1`), `Arguments` is a list of `ArgumentName-Value` pairs, and `Result` is unified with the tool result. The result must be one of: `text(Atom)` for a text result, `error(Atom)` for an error result, `results(List)` for a list of content items where each item is `text(Atom)`, `error(Atom)`, `resource_link(URI, Name)`, or `resource_link(URI, Name, Description, MimeType)`, `structured(StructuredContent)` for structured output with auto-generated text, or `structured(ContentItems, StructuredContent)` for structured output with explicit content items. If this predicate is not defined for a tool, the MCP server will use auto-dispatch: it calls the tool predicate as a message to the implementing object, collects output arguments, and returns them as a text result.

Compilation flags:

```
static
```

Template:

```
tool_call(Name,Arguments,Result)
```

Mode and number of proofs:

```
tool_call(+atom,+list(pair),--compound) - one
```

---

tool_call/4

Handles a tool call with elicitation support under the 2025-06-18 adapter only. Same as `tool_call/3` but receives an `Elicit` closure as the third argument. The closure can be called as `call(Elicit, Message, Schema, Answer)` where `Message` is a prompt text (an atom), `Schema` is a curly-term JSON Schema describing the requested input (e.g., `{type-object, properties-{answer-{type-string, enum-[yes, no]}}, required-[answer]}`), and `Answer` is unified with `accept(Content)` (where `Content` is the user response as a curly-term), `decline`, or `cancel`. Only available when the application declares `elicitation` in its `capabilities/1` and the client advertises elicitation support during initialization. Falls back to `tool_call/3` and then auto-dispatch if not defined. The 2026-07-28 adapter never calls this predicate; use `tool_call_round/4` from `mcp_multiround_protocol` instead.

Compilation flags:

```
static
```

Template:

```
tool_call(Name,Arguments,Elicit,Result)
```

Mode and number of proofs:

```
tool_call(+atom,+list(pair),+callable,--compound) - one
```

---

output_schema/2

Returns the JSON output schema for the given tool name. Optional; when defined, the schema overrides the one inferred from the tool predicate `info/2` and `mode/2` directives. The tool can return `structured(StructuredContent)` or `structured(ContentItems, StructuredContent)` results. The schema must be a curly-term following JSON Schema format (e.g. `{type-object, properties-{temperature-{type-number}}, required-[temperature]}`).

Compilation flags:

```
static
```

Template:

```
output_schema(Name,Schema)
```

Mode and number of proofs:

```
output_schema(+atom,-compound) - zero_or_one
```

---

input_schema/2

Returns the JSON input schema for the given tool name. Optional; when defined, the schema overrides the one inferred from the tool predicate `info/2` and `mode/2` directives. The schema must be a curly-term following JSON Schema format (e.g. `{type-object, properties-{temperature-{type-number}}, required-[temperature]}`).

Compilation flags:
`static`

Template:
`input_schema(Name,Schema)`
Mode and number of proofs:
`input_schema(+atom,-compound)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.214.17 mcp_ui_protocol

Optional MCP Apps (`io.modelcontextprotocol/ui`) metadata for tools and resources. Applications that provide interactive UI resources declare `ui` in `capabilities/1` (and typically `resources`), implement `mcp_resource_protocol` for `ui://` HTML resources, and may implement these predicates to attach tool and resource UI metadata. See the https://modelcontextprotocol.io/extensions/apps/overview spec for details.

Availability:
`logtalk_load(mcp_server(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-30

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *tool_ui/2*
  - *resource_ui_meta/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

tool_ui/2

Optional. UI metadata for a tool. `Options` is a list of `resource_uri(URI)` and/or `visibility(List)` terms. `URI` must be a `ui://` resource served by the application. `List` is a list of `model` and/or `app` (default when omitted on the wire is both). App-only tools use `visibility([app])` so hosts hide them from the model while still allowing iframe-proxied `tools/call`.

Compilation flags:

`static`

Template:

`tool_ui(ToolName,Options)`

Mode and number of proofs:

`tool_ui(?atom,-list)` - `zero_or_more`

---

resource_ui_meta/2

Optional. UI metadata for a `ui://` resource as a curly-term matching the Apps `_meta.ui` object (`csp`, `permissions`, `domain`, `prefersBorder`). Attached when serializing resource descriptors and may be used for `resources/read` content.

Compilation flags:

`static`

Template:

`resource_ui_meta(URI,Meta)`

Mode and number of proofs:

`resource_ui_meta(?atom,-compound)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

mcp_tool_protocol, *mcp_resource_protocol*, *mcp_server*

1.215 memcached

object

1.215.1 `memcached`

Portable Memcached client implementing the text (ASCII) protocol. Uses the sockets library for TCP communication.

Availability:

```
logtalk_load(memcached(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-11-06

Compilation flags:

```
static, context_switching_calls
```

Uses:

list
socket
user

Remarks:

- Supported backends: ECLiPSe, GNU Prolog, SICStus Prolog, Shumway Prolog, SWI-Prolog, and Trealla Prolog (same as the sockets library).
- Protocol version: Implements the Memcached text (ASCII) protocol as documented in the official protocol.txt specification.
- Connection: The default Memcached port is 11211. Connections are represented as opaque handles that should be passed to all other predicates.
- Keys: Keys are atoms up to 250 characters. They must not include control characters or whitespace.
- Flags: Flags are 32-bit unsigned integers stored alongside the data, opaque to the server.
- Expiration: Expiration time in seconds. 0 means never expire. Values over 30 days (2592000) are treated as Unix timestamps.
- Storage commands: Supports set, add, replace, append, prepend, and cas (check-and-set).
- Retrieval commands: Supports get, gets (with CAS token), gat, and gats (get-and-touch).

- Other commands: Supports delete, incr/decr, touch, flush_all, version, stats, and quit.

Inherited public predicates:

(none)

- *Public predicates*
  - *connect/3*
  - *connect/2*
  - *disconnect/1*
  - *set/5*
  - *set/3*
  - *add/5*
  - *replace/5*
  - *append/3*
  - *prepend/3*
  - *cas/6*
  - *get/3*
  - *get/4*
  - *gets/4*
  - *gets/5*
  - *mget/3*
  - *delete/2*
  - *incr/4*
  - *decr/4*
  - *touch/3*
  - *gat/4*
  - *gats/5*
  - *flush_all/1*
  - *flush_all/2*
  - *version/2*
  - *stats/2*
  - *stats/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

connect/3

Connects to a Memcached server at the given host and port. Returns a connection handle for subsequent operations.

Compilation flags:

```
static
```

Template:

```
connect(Host,Port,Connection)
```

Mode and number of proofs:

`connect(+atom,+integer,--compound)` - `one_or_error`

Exceptions:

Connection refused or network error:

`memcached_error(connection_failed)`

---

connect/2

Connects to a Memcached server at the given host on the default port (11211). Returns a connection handle for subsequent operations.

Compilation flags:

`static`

Template:

`connect(Host,Connection)`

Mode and number of proofs:

`connect(+atom,--compound)` - `one_or_error`

Exceptions:

Connection refused or network error:

`memcached_error(connection_failed)`

---

disconnect/1

Disconnects from the Memcached server. Sends a quit command before closing.

Compilation flags:

`static`

Template:

`disconnect(Connection)`

Mode and number of proofs:

`disconnect(+compound)` - `one`

---

set/5

Stores the data unconditionally. Overwrites any existing data for the key.

Compilation flags:

`static`

Template:

```
set(Connection,Key,Value,Flags,ExpTime)
```

Mode and number of proofs:

```
set(+compound,+atom,+atom,+integer,+integer) - one_or_error
```

Exceptions:

Storage failed:

```
memcached_error(not_stored)
```

Network error:

```
memcached_error(Error)
```

---

set/3

Stores the data unconditionally with default flags (0) and no expiration (0).

Compilation flags:

```
static
```

Template:

```
set(Connection,Key,Value)
```

Mode and number of proofs:

```
set(+compound,+atom,+atom) - one_or_error
```

Exceptions:

Storage failed:

```
memcached_error(not_stored)
```

Network error:

```
memcached_error(Error)
```

---

add/5

Stores the data only if the key does not already exist.

Compilation flags:

```
static
```

Template:

```
add(Connection,Key,Value,Flags,ExpTime)
```

Mode and number of proofs:

```
add(+compound,+atom,+atom,+integer,+integer) - one_or_error
```

Exceptions:

Key already exists:

```
memcached_error(not_stored)
```

Network error:

```
memcached_error(Error)
```

---

replace/5

Stores the data only if the key already exists.

Compilation flags:
`static`

Template:
`replace(Connection,Key,Value,Flags,ExpTime)`
Mode and number of proofs:
`replace(+compound,+atom,+atom,+integer,+integer) - one_or_error`

Exceptions:
Key does not exist:
`memcached_error(not_stored)`
Network error:
`memcached_error(Error)`

---

append/3

Appends the data to the end of an existing item's data.

Compilation flags:
`static`

Template:
`append(Connection,Key,Value)`
Mode and number of proofs:
`append(+compound,+atom,+atom) - one_or_error`

Exceptions:
Key does not exist:
`memcached_error(not_stored)`
Network error:
`memcached_error(Error)`

---

prepend/3

Prepends the data to the beginning of an existing item's data.

Compilation flags:
`static`

Template:
`prepend(Connection,Key,Value)`

Mode and number of proofs:

```
prepend(+compound,+atom,+atom) - one_or_error
```

Exceptions:

Key does not exist:

```
memcached_error(not_stored)
```

Network error:

```
memcached_error(Error)
```

---

cas/6

Stores the data only if no one else has updated it since the given CAS unique value was obtained (via gets/3).

Compilation flags:

```
static
```

Template:

```
cas(Connection,Key,Value,Flags,ExpTime,CasUnique)
```

Mode and number of proofs:

```
cas(+compound,+atom,+atom,+integer,+integer,+integer) - one_or_error
```

Exceptions:

CAS value mismatch (item modified by another client):

```
memcached_error(exists)
```

Key does not exist:

```
memcached_error(not_found)
```

Network error:

```
memcached_error(Error)
```

---

get/3

Retrieves the value associated with the key. Fails if the key is not found.

Compilation flags:

```
static
```

Template:

```
get(Connection,Key,Value)
```

Mode and number of proofs:

```
get(+compound,+atom,-atom) - zero_or_one_or_error
```

Exceptions:

Network or stream I/O error while sending or reading the `get` command:

```
memcached_error(Error)
```

---

get/4

Retrieves the value and flags associated with the key. Fails if the key is not found.

Compilation flags:
`static`

Template:
`get(Connection,Key,Value,Flags)`
Mode and number of proofs:
`get(+compound,+atom,-atom,-integer)` - `zero_or_one_or_error`

Exceptions:
Network or stream I/O error while sending or reading the `get` command:
`memcached_error(Error)`

---

gets/4

Retrieves the value and CAS unique token for the key. Fails if the key is not found. The CAS value is used with the cas/6 predicate.

Compilation flags:
`static`

Template:
`gets(Connection,Key,Value,CasUnique)`
Mode and number of proofs:
`gets(+compound,+atom,-atom,-integer)` - `zero_or_one_or_error`

Exceptions:
Network or stream I/O error while sending or reading the `gets` command:
`memcached_error(Error)`

---

gets/5

Retrieves the value, flags, and CAS unique token for the key. Fails if the key is not found.

Compilation flags:
`static`

Template:
`gets(Connection,Key,Value,Flags,CasUnique)`
Mode and number of proofs:
`gets(+compound,+atom,-atom,-integer,-integer)` - `zero_or_one_or_error`

Exceptions:

Network or stream I/O error while sending or reading the `gets` command:
`memcached_error(Error)`

---

mget/3

Retrieves multiple keys at once. Returns a list of `item(Key, Value, Flags)` terms for found keys.

Compilation flags:
`static`

Template:
`mget(Connection,Keys,Items)`
Mode and number of proofs:
`mget(+compound,+list(atom),-list(compound))` - `one_or_error`

Exceptions:
Network or stream I/O error while sending or reading the multi-key retrieval command:
`memcached_error(Error)`

---

delete/2

Deletes the item with the given key. Fails if the key is not found.

Compilation flags:
`static`

Template:
`delete(Connection,Key)`
Mode and number of proofs:
`delete(+compound,+atom)` - `zero_or_one_or_error`

Exceptions:
Network or stream I/O error while sending or reading the `delete` command:
`memcached_error(Error)`

---

incr/4

Increments the numeric value of the given key by the specified amount. Returns the new value. The item must already exist and contain a decimal representation of a 64-bit unsigned integer. Fails if the key is not found.

Compilation flags:
`static`

Template:

```
incr(Connection,Key,Amount,NewValue)
```

Mode and number of proofs:

```
incr(+compound,+atom,+integer,-integer) - zero_or_one_or_error
```

Exceptions:

Network or stream I/O error while sending or reading the `incr` command:

```
memcached_error(Error)
```

---

decr/4

Decrements the numeric value of the given key by the specified amount. Returns the new value. Underflow is caught: decrementing below 0 yields 0. Fails if the key is not found.

Compilation flags:

```
static
```

Template:

```
decr(Connection,Key,Amount,NewValue)
```

Mode and number of proofs:

```
decr(+compound,+atom,+integer,-integer) - zero_or_one_or_error
```

Exceptions:

Network or stream I/O error while sending or reading the `decr` command:

```
memcached_error(Error)
```

---

touch/3

Updates the expiration time of the given key without fetching the data. Fails if the key is not found.

Compilation flags:

```
static
```

Template:

```
touch(Connection,Key,ExpTime)
```

Mode and number of proofs:

```
touch(+compound,+atom,+integer) - zero_or_one_or_error
```

Exceptions:

Network or stream I/O error while sending or reading the `touch` command:

```
memcached_error(Error)
```

---

gat/4

Gets the value of the key and updates its expiration time. Fails if the key is not found.

Compilation flags:
`static`

Template:
`gat(Connection,Key,ExpTime,Value)`
Mode and number of proofs:
`gat(+compound,+atom,+integer,-atom)` - `zero_or_one_or_error`

Exceptions:
Network or stream I/O error while sending or reading the `gat` command:
`memcached_error(Error)`

---

gats/5

Gets the value and CAS unique token of the key and updates its expiration time. Fails if the key is not found.

Compilation flags:
`static`

Template:
`gats(Connection,Key,ExpTime,Value,CasUnique)`
Mode and number of proofs:
`gats(+compound,+atom,+integer,-atom,-integer)` - `zero_or_one_or_error`

Exceptions:
Network or stream I/O error while sending or reading the `gats` command:
`memcached_error(Error)`

---

flush_all/1

Invalidates all existing items immediately.

Compilation flags:
`static`

Template:
`flush_all(Connection)`
Mode and number of proofs:
`flush_all(+compound)` - `one_or_error`

Exceptions:

Network or stream I/O error while sending or reading the `flush_all` command:
`memcached_error(Error)`
The server replies with an error to the `flush_all` command:
`memcached_error(server_error(Message))`

`flush_all/2`

Invalidates all existing items after the specified number of seconds.

Compilation flags:
`static`

Template:
`flush_all(Connection,Delay)`
Mode and number of proofs:
`flush_all(+compound,+integer)` - `one_or_error`

Exceptions:
Network or stream I/O error while sending or reading the delayed `flush_all` command:
`memcached_error(Error)`
The server replies with an error to the delayed `flush_all` command:
`memcached_error(server_error(Message))`

`version/2`

Returns the version string of the Memcached server.

Compilation flags:
`static`

Template:
`version(Connection,Version)`
Mode and number of proofs:
`version(+compound,-atom)` - `one_or_error`

Exceptions:
Network or stream I/O error while sending or reading the `version` command:
`memcached_error(Error)`
The server replies with an unexpected or error response to the `version` command:
`memcached_error(server_error(Message))`

---

stats/2

Returns general-purpose statistics as a list of `stat(Name, Value)` terms.

Compilation flags:
`static`

Template:
`stats(Connection,Stats)`
Mode and number of proofs:
`stats(+compound,-list(compound))` - `one_or_error`

Exceptions:
Network or stream I/O error while sending or reading the `stats` command:
`memcached_error(Error)`

---

stats/3

Returns statistics for the given argument (e.g. `items`, `slabs`, `sizes`) as a list of `stat(Name, Value)` terms.

Compilation flags:
`static`

Template:
`stats(Connection,Argument,Stats)`
Mode and number of proofs:
`stats(+compound,+atom,-list(compound))` - `one_or_error`

Exceptions:
Network or stream I/O error while sending or reading the parameterized `stats` command:
`memcached_error(Error)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.216 message_pack

object

1.216.1 message_pack(StringRepresentation)

- `StringRepresentation` - Text representation to be used when decoding MessagePack strings. Possible values are `atom` (default), `chars`, and `codes`.

MessagePack format exporter and importer.

Availability:
`logtalk_load(message_pack(loader))`

Author: Paulo Moura
Version: 1:1:2
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Uses:
ieee_754(Precision,ByteOrder,NaNRepresentation)
list
type
utf_8

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a list of bytes in the MessagePack format returning the corresponding term representation. Throws an error when parsing is not possible (usually due to an invalid byte sequence).

Compilation flags:

`static`

Template:

`parse(Bytes,Term)`

Mode and number of proofs:

`parse(@list(byte),-ground)` - `one_or_error`

Exceptions:

`Bytes` is not a ground term:

`instantiation_error`

`Bytes` is ground but not a valid MessagePack byte sequence:

`domain_error(message_pack_byte_sequence,Bytes)`

---

generate/2

Generates a list of bytes in the MessagePack format representing the given term. Throws an error when generating is not possible (usually due to a term with no corresponding MessagePack representation).

Compilation flags:

`static`

Template:

`generate(Term,Bytes)`

Mode and number of proofs:

`generate(@ground,-list(byte))` - `one_or_error`

Exceptions:

`Term` is a variable:

`instantiation_error`

`Term` has no corresponding MessagePack representation:

`domain_error(message_pack_term,Term)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.217 meta

object

1.217.1 `meta`

Some useful meta-predicates.

Availability:
`logtalk_load(meta(loader))`

Author: Paulo Moura
Version: 5:2:0
Date: 2016-10-06

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *metap*
Aliases:
`metap map/2` as `succeeds/2`
`metap map/2` as `maplist/2`
`metap map/3` as `maplist/3`
`metap map/4` as `maplist/4`
`metap map/5` as `maplist/5`
`metap map/6` as `maplist/6`
`metap map/7` as `maplist/7`
`metap map/8` as `maplist/8`
`metap include/3` as `filter/3`
`metap fold_left/4` as `foldl/4`
`metap fold_left_1/3` as `foldl1/3`
`metap fold_right/4` as `foldr/4`
`metap fold_right_1/3` as `foldr1/3`
`metap scan_left/4` as `scanl/4`
`metap scan_left_1/3` as `scanl1/3`
`metap scan_right/4` as `scanr/4`
`metap scan_right_1/3` as `scanr1/3`

Remarks:

(none)

Inherited public predicates:

exclude/3 findall_member/4 findall_member/5 fold_left/4 fold_left_1/3 fold_right/4 fold_right_1/3 include/3 map/2 map/3 map/4 map/5 map/6 map/7 map/8 map_reduce/5 partition/4 partition/6 scan_left/4 scan_left_1/3 scan_right/4 scan_right_1/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

meta_compiler

protocol

1.217.2 `metap`

Useful meta-predicates protocol.

Availability:

`logtalk_load(meta(loader))`

Author: Paulo Moura
Version: 6:1:0
Date: 2015-12-23

Compilation flags:

`static`

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *include/3*
  - *exclude/3*
  - *findall_member/4*
  - *findall_member/5*
  - *partition/4*
  - *partition/6*
  - *fold_left/4*
  - *fold_left_1/3*
  - *scan_left/4*
  - *scan_left_1/3*
  - *fold_right/4*
  - *fold_right_1/3*
  - *scan_right/4*
  - *scan_right_1/3*
  - *map/2*
  - *map/3*
  - *map/4*
  - *map/5*
  - *map/6*
  - *map/7*
  - *map/8*
  - *map_reduce/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

include/3

Returns a list of all list elements that satisfy a predicate.

Compilation flags:
`static`

Template:
`include(Closure,List,Included)`

Meta-predicate template:

```
include(1,*,*)
```

Mode and number of proofs:

```
include(+callable,+list,-list) - one
```

---

exclude/3

Returns a list of all list elements that fail to satisfy a predicate.

Compilation flags:

```
static
```

Template:

```
exclude(Closure,List,Excluded)
```

Meta-predicate template:

```
exclude(1,*,*)
```

Mode and number of proofs:

```
exclude(+callable,+list,-list) - one
```

---

findall_member/4

Finds all members of a list that satisfy a given test.

Compilation flags:

```
static
```

Template:

```
findall_member(Member,List,Test,Result)
```

Meta-predicate template:

```
findall_member(*,*,0,*)
```

Mode and number of proofs:

```
findall_member(@term,+list,@callable,-list) - one
```

---

findall_member/5

Finds all members of a list that satisfy a given test appending the given tail to the result.

Compilation flags:

```
static
```

Template:

```
findall_member(Member,List,Test,Result,Tail)
```

Meta-predicate template:

```
findall_member(*,*,0,*,*)
```

Mode and number of proofs:

```
findall_member(@term,+list,@callable,-list,+list) - one
```

---

partition/4

Partition a list of elements in two lists using a predicate.

Compilation flags:

```
static
```

Template:

```
partition(Closure,List,Included,Excluded)
```

Meta-predicate template:

```
partition(1,*,*,*)
```

Mode and number of proofs:

```
partition(+callable,+list,-list,-list) - one
```

---

partition/6

Partitions a list in lists with values less, equal, and greater than a given value using a comparison predicate with the same argument order as `compare/3`.

Compilation flags:

```
static
```

Template:

```
partition(Closure,List,Value,Less,Equal,Greater)
```

Meta-predicate template:

```
partition(3,*,*,*,*,*)
```

Mode and number of proofs:

```
partition(+callable,+list,@term,-list,-list,-list) - one
```

---

fold_left/4

List folding (left associative). Closure is extended with three arguments: accumulator, list element, and updated accumulator.

Compilation flags:

```
static
```

Template:

```
fold_left(Closure,Accumulator,List,Result)
```

Meta-predicate template:

```
fold_left(3,*,*,*)
```

Mode and number of proofs:

```
fold_left(+callable,?term,+list,?term) - zero_or_more
```

---

fold_left_1/3

List folding (left associative). Closure is extended with three arguments: accumulator, list element, and updated accumulator. The initial value of the accumulator is the list first element. Fails for empty lists.

Compilation flags:

```
static
```

Template:

```
fold_left_1(Closure,List,Result)
```

Meta-predicate template:

```
fold_left_1(3,*,*)
```

Mode and number of proofs:

```
fold_left_1(+callable,+list,?term) - zero_or_more
```

---

scan_left/4

List scanning (left associative). Closure is extended with three arguments: accumulator, list element, and updated accumulator.

Compilation flags:

```
static
```

Template:

```
scan_left(Closure,Accumulator,List,Results)
```

Meta-predicate template:

```
scan_left(3,*,*,*)
```

Mode and number of proofs:

```
scan_left(+callable,?term,+list,?list) - zero_or_more
```

---

scan_left_1/3

List scanning (left associative). Closure is extended with three arguments: accumulator, list element, and updated accumulator. The accumulator is initialized with the list first element. Fails for empty lists.

Compilation flags:

```
static
```

Template:

```
scan_left_1(Closure,List,Results)
```

Meta-predicate template:

```
scan_left_1(3,*,*)
```

Mode and number of proofs:

```
scan_left_1(+callable,+list,?list) - zero_or_more
```

---

fold_right/4

List folding (right associative). Closure is extended with three arguments: list element, accumulator, and updated accumulator.

Compilation flags:

```
static
```

Template:

```
fold_right(Closure,Accumulator,List,Result)
```

Meta-predicate template:

```
fold_right(3,*,*,*)
```

Mode and number of proofs:

```
fold_right(+callable,?term,+list,?term) - zero_or_more
```

---

fold_right_1/3

List folding (right associative). Closure is extended with three arguments: list element, accumulator, and updated accumulator. The initial value of the accumulator is the list first element. Fails for empty lists.

Compilation flags:

```
static
```

Template:

```
fold_right_1(Closure,List,Result)
```

Meta-predicate template:

```
fold_right_1(3,*,*)
```

Mode and number of proofs:

```
fold_right_1(+callable,+list,?term) - zero_or_more
```

---

scan_right/4

List scanning (right associative). Closure is extended with three arguments: list element, accumulator, and updated accumulator.

Compilation flags:
`static`

Template:
`scan_right(Closure,Accumulator,List,Results)`
Meta-predicate template:
`scan_right(3,*,*,*)`
Mode and number of proofs:
`scan_right(+callable,?term,+list,?list)` - `zero_or_more`

---

scan_right_1/3

List scanning (right associative). Closure is extended with three arguments: list element, accumulator, and updated accumulator. The accumulator is initialized with the list first element. Fails for empty lists.

Compilation flags:
`static`

Template:
`scan_right_1(Closure,List,Results)`
Meta-predicate template:
`scan_right_1(3,*,*)`
Mode and number of proofs:
`scan_right_1(+callable,+list,?list)` - `zero_or_more`

---

map/2

True if the predicate succeeds for each list element.

Compilation flags:
`static`

Template:
`map(Closure,List)`
Meta-predicate template:
`map(1,*)`
Mode and number of proofs:
`map(+callable,?list)` - `zero_or_more`

---

map/3

List mapping predicate taken arguments from two lists of elements.

Compilation flags:
`static`

Template:
`map(Closure,List1,List2)`
Meta-predicate template:
`map(2,*,*)`
Mode and number of proofs:
`map(+callable,?list,?list)` - `zero_or_more`

---

map/4

List mapping predicate taken arguments from three lists of elements.

Compilation flags:
`static`

Template:
`map(Closure,List1,List2,List3)`
Meta-predicate template:
`map(3,*,*,*)`
Mode and number of proofs:
`map(+callable,?list,?list,?list)` - `zero_or_more`

---

map/5

List mapping predicate taken arguments from four lists of elements.

Compilation flags:
`static`

Template:
`map(Closure,List1,List2,List3,List4)`
Meta-predicate template:
`map(4,*,*,*,*)`
Mode and number of proofs:
`map(+callable,?list,?list,?list,?list)` - `zero_or_more`

---

map/6

List mapping predicate taken arguments from five lists of elements.

Compilation flags:
`static`

Template:
`map(Closure,List1,List2,List3,List4,List5)`
Meta-predicate template:
`map(5,*,*,*,*,*)`
Mode and number of proofs:
`map(+callable,?list,?list,?list,?list,?list)` - `zero_or_more`

---

map/7

List mapping predicate taken arguments from six lists of elements.

Compilation flags:
`static`

Template:
`map(Closure,List1,List2,List3,List4,List5,List6)`
Meta-predicate template:
`map(6,*,*,*,*,*,*)`
Mode and number of proofs:
`map(+callable,?list,?list,?list,?list,?list,?list)` - `zero_or_more`

---

map/8

List mapping predicate taken arguments from seven lists of elements.

Compilation flags:
`static`

Template:
`map(Closure,List1,List2,List3,List4,List5,List6,List7)`
Meta-predicate template:
`map(7,*,*,*,*,*,*,*)`
Mode and number of proofs:
`map(+callable,?list,?list,?list,?list,?list,?list,?list)` - `zero_or_more`

---

map_reduce/5

Map a list and apply a fold left (reduce) to the resulting list.

Compilation flags:
`static`

Template:
`map_reduce(Map,Reduce,Accumulator,List,Result)`
Meta-predicate template:
`map_reduce(2,3,*,*,*)`
Mode and number of proofs:
`map_reduce(+callable,+callable,+term,?list,?term) - zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

meta

1.218 meta_compiler

object

1.218.1 meta_compiler

Compiler for the `meta` object meta-predicates. Generates auxiliary predicates in order to avoid meta-call overheads.

Availability:
`logtalk_load(meta_compiler(loader))`

Author: Paulo Moura
Version: 0:16:1
Date: 2026-04-22

Compilation flags:
`static, context_switching_calls`

Implements:

`public` *expanding*

Uses:

gensym
list
logtalk
user

Remarks:

- Usage: Compile source files with calls to the `meta` object meta-predicates using the compiler option `hook(meta_compiler)`.

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *generated_predicate_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

generated_predicate_/1

Table of generated auxiliary predicates.

Compilation flags:

`dynamic`

Template:

`generated_predicate_(Predicate)`

Mode and number of proofs:

`generated_predicate_(?predicate_indicator)` - `zero_or_more`

Operators

(none)

See also

meta

1.219 metagol

object

1.219.1 `metagol`

Inductive logic programming (ILP) system based on meta-interpretive learning.

Availability:
`logtalk_load(metagol(loader))`

Author: Metagol authors; adapted to Logtalk by Paulo Moura.
Version: 0:25:0
Date: 2026-08-10

Copyright: Copyright 2016 Metagol authors; Copyright 2018-2024 Paulo Moura
License: BSD-3-Clause

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Provides:
logtalk::message_tokens//2
logtalk::message_prefix_stream/4
Uses:
coroutining
integer
list
logtalk
meta
timeout

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
  - *learn/3*
  - *learn/2*
  - *learn_seq/2*
  - *learn_with_timeout/4*
  - *program_to_clauses/2*
  - *pprint/1*
  - *metarule/6*
  - *head_pred/1*
  - *ibk/3*
  - *func_test/3*
  - *body_pred/1*
  - *functional/0*
  - *min_clauses/1*
  - *max_clauses/1*
  - *max_inv_preds/1*
  - *metarule_next_id/1*
  - *timeout/1*
- *Protected predicates*
  - *pprint_clause/1*
  - *compiled_pred_call/2*
  - *body_pred_call/2*
  - *type/3*
- *Private predicates*
- *Operators*

Public predicates

learn/3

Learns from a set of positive examples and a set of negative examples and returns the learned program.

Compilation flags:

```
static
```

Template:

```
learn(PositiveExamples,NegativeExamples,Program)
```

Mode and number of proofs:

```
learn(@list(example),@list(example),-list(term)) - zero_or_more
```

learn/2

Learns from a set of positive examples and a set of negative examples and pretty prints the learned program.

Compilation flags:
`static`

Template:
`learn(PositiveExamples,NegativeExamples)`
Mode and number of proofs:
`learn(@list(example),@list(example))` - `zero_or_more`

---

learn_seq/2

Learns from a sequence of examples represented as a list of PositiveExamples/NegativeExamples elements and returns the learned program.

Compilation flags:
`static`

Template:
`learn_seq(Examples,Program)`
Mode and number of proofs:
`learn_seq(@list(example),-list(clause))` - `zero_or_one`

---

learn_with_timeout/4

Learns from a set of positive examples and a set of negative examples and returns the learned program constrained by the given timeout or its default value.

Compilation flags:
`static`

Template:
`learn_with_timeout(PositiveExamples,NegativeExamples,Program,Timeout)`
Mode and number of proofs:
`learn_with_timeout(@list(example),@list(example),-list(term),+number)` - `zero_or_one_or_error`
`learn_with_timeout(@list(example),@list(example),-list(term),-number)` - `zero_or_one_or_error`

Exceptions:
Learning does not complete in the allowed time:
`timeout(learn(PositiveExamples,NegativeExamples,Program))`

---

program_to_clauses/2

Converts a learned program into a list of clauses.

Compilation flags:
```
static
```

Template:
```
program_to_clauses(Program,Clauses)
```
Mode and number of proofs:
```
program_to_clauses(@list(term),-list(clause)) - one
```

---

pprint/1

Pretty prints a learned program.

Compilation flags:
```
static
```

Template:
```
pprint(Program)
```
Mode and number of proofs:
```
pprint(@list(term)) - one
```

---

metarule/6

Compiled metarule, generated from a user-defined metarule/3 or metarule/4 clause, relating a name, its substitutions, a head atom, a body, a recursion flag, and the current call path.

Compilation flags:
```
static
```

Template:
```
metarule(Name,Subs,Head,Body,Recursive,Path)
```
Mode and number of proofs:
```
metarule(?atom,?list,?callable,?list,?atom,?list) - zero_or_more
```

---

head_pred/1

Declares a predicate indicator that can be used in the head of a learned clause.

Compilation flags:
`static`

Template:
`head_pred(PredicateIndicator)`
Mode and number of proofs:
`head_pred(?predicate_indicator)` - `zero_or_more`

---

ibk/3

Interpreted background knowledge clause, generated from a user-defined ibk/2 fact or rule, relating a head atom (as a list) to a body (as a list of atoms) and the current call path.

Compilation flags:
`static`

Template:
`ibk(Head,Body,Path)`
Mode and number of proofs:
`ibk(?list,?list,?list)` - `zero_or_more`

---

func_test/3

Relates an atom to a test atom and a condition used to check that the learned program is functional.

Compilation flags:
`static`

Template:
`func_test(Atom,TestAtom,Condition)`
Mode and number of proofs:
`func_test(@callable,-callable,-callable)` - `zero_or_more`

---

body_pred/1

Declares a predicate indicator that can be used in the body of a learned clause.

Compilation flags:
`dynamic`

Template:
`body_pred(PredicateIndicator)`
Mode and number of proofs:
`body_pred(?predicate_indicator)` - `zero_or_more`

---

functional/0

True when the learned program is required to be functional (deterministic).

Compilation flags:
`dynamic`

Mode and number of proofs:
`functional` - `zero_or_one`

---

min_clauses/1

Minimum number of clauses to try when searching for a learned program.

Compilation flags:
`dynamic`

Template:
`min_clauses(MinClauses)`
Mode and number of proofs:
`min_clauses(?integer)` - `zero_or_one`

---

max_clauses/1

Maximum number of clauses to try when searching for a learned program.

Compilation flags:
`dynamic`

Template:

```
max_clauses(MaxClauses)
```
Mode and number of proofs:
```
max_clauses(?integer) - zero_or_one
```

---

max_inv_preds/1

Maximum number of invented predicate symbols allowed per learning task.

Compilation flags:
```
dynamic
```

Template:
```
max_inv_preds(MaxInventedPredicates)
```
Mode and number of proofs:
```
max_inv_preds(?integer) - zero_or_one
```

---

metarule_next_id/1

Next identifier to use when generating a name for an anonymous metarule.

Compilation flags:
```
dynamic
```

Template:
```
metarule_next_id(Id)
```
Mode and number of proofs:
```
metarule_next_id(?integer) - zero_or_one
```

---

timeout/1

Default timeout, in seconds, used when learning a program.

Compilation flags:
```
dynamic
```

Template:
```
timeout(Timeout)
```
Mode and number of proofs:
```
timeout(?number) - zero_or_one
```

Protected predicates

pprint_clause/1

Pretty prints a single learned clause. Can be redefined to customize the printing of learned clauses.

Compilation flags:
```
static
```

Template:
```
pprint_clause(Clause)
```
Mode and number of proofs:
```
pprint_clause(@clause) - one
```

---

compiled_pred_call/2

Asserted clause used to call a compiled (pre-existing) predicate given its name and list of arguments.

Compilation flags:
```
dynamic
```

Template:
```
compiled_pred_call(Predicate,Arguments)
```
Mode and number of proofs:
```
compiled_pred_call(?atom,?list) - zero_or_more
```

---

body_pred_call/2

Asserted clause used to call a background knowledge predicate given its name and list of arguments.

Compilation flags:
```
dynamic
```

Template:
```
body_pred_call(Predicate,Arguments)
```
Mode and number of proofs:
```
body_pred_call(?atom,?list) - zero_or_more
```

---

type/3

Asserted fact classifying a predicate, given by its name and arity, by its role (head_pred, body_pred, ibk_head_pred, or compiled_pred).

Compilation flags:
`dynamic`

Template:
`type(Name,Arity,Type)`
Mode and number of proofs:
`type(?atom,?integer,?atom)` - `zero_or_more`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.219.2 metagol_example_protocol

Convenient learning predicates for use in examples and unit tests.

Availability:
`logtalk_load(metagol(loader))`

Author: Paulo Moura.
Version: 0:1:1
Date: 2024-03-15

License: BSD-3-Clause

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/1*
  - *learn/0*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/1

Learns and returns set of clauses.

Compilation flags:
`static`

Template:
`learn(Clauses)`

Mode and number of proofs:
`learn(-list(clause))` - `zero_or_more`

---

learn/0

Learns and prints a set of clauses.

Compilation flags:
`static`

Mode and number of proofs:
`learn` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.220 mime_types

object

1.220.1 `mime_types`

MIME type registry and convenience predicates for mapping file names and URLs to media types.

Availability:
`logtalk_load(mime_types(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-06-01

Compilation flags:
`static, context_switching_calls`

Uses:
list
os
reader
string(Representation)

Remarks:

- Default behavior: Convenience predicates default to lenient lookup by consulting both built-in standard mappings and built-in common mappings.
- Path decomposition: File path decomposition uses the `os` library `decompose_file_name/4` predicate.
- External overlays: Additional `mime.types`-style files can be loaded into the in-memory registry using the `load/1-2` predicates.

Inherited public predicates:
(none)

- *Public predicates*
  - *guess_type/3*
  - *guess_type/4*
  - *guess_file_type/3*
  - *guess_file_type/4*
  - *extension_type/2*
  - *extension_type/3*
  - *guess_extension/2*
  - *guess_extension/3*

      - *guess_all_extensions/2*
      - *guess_all_extensions/3*
      - *add_type/2*
      - *add_type/3*
      - *read_mime_types/2*
      - *load/1*
      - *load/2*
      - *reset/0*
      - *suffix_alias/2*
      - *encoding_suffix/2*
  - *Protected predicates*
  - *Private predicates*
      - *runtime_type_/3*
  - *Operators*

Public predicates

guess_type/3

Guesses the MIME type and content encoding for a URL or file name using lenient lookup. Unknown values are returned as the empty atom.

Compilation flags:

```
static
```

Template:

```
guess_type(Resource,Type,Encoding)
```

Mode and number of proofs:

```
guess_type(+atom,-atom,-atom) - one
```

---

guess_type/4

Guesses the MIME type and content encoding for a URL or file name. When `Strict` is `true`, only built-in standard mappings and strict runtime overlays are consulted.

Compilation flags:

```
static
```

Template:

```
guess_type(Resource,Type,Encoding,Strict)
```

Mode and number of proofs:

```
guess_type(+atom,-atom,-atom,+boolean) - one
```

---

guess_file_type/3

Guesses the MIME type and content encoding for a file path using lenient lookup. Unknown values are returned as the empty atom.

Compilation flags:
```
static
```

Template:
```
guess_file_type(Path,Type,Encoding)
```
Mode and number of proofs:
```
guess_file_type(+atom,-atom,-atom) - one
```

---

guess_file_type/4

Guesses the MIME type and content encoding for a file path.

Compilation flags:
```
static
```

Template:
```
guess_file_type(Path,Type,Encoding,Strict)
```
Mode and number of proofs:
```
guess_file_type(+atom,-atom,-atom,+boolean) - one
```

---

extension_type/2

Returns the MIME type associated with a file extension using lenient lookup.

Compilation flags:
```
static
```

Template:
```
extension_type(Extension,Type)
```
Mode and number of proofs:
```
extension_type(+atom,-atom) - zero_or_one
```

---

extension_type/3

Returns the MIME type associated with a file extension.

Compilation flags:
```
static
```

Template:
```
extension_type(Extension,Type,Strict)
```
Mode and number of proofs:
```
extension_type(+atom,-atom,+boolean) - zero_or_one
```

---

guess_extension/2

Returns the preferred file extension associated with a MIME type using lenient lookup.

Compilation flags:
```
static
```

Template:
```
guess_extension(Type,Extension)
```
Mode and number of proofs:
```
guess_extension(+atom,-atom) - zero_or_one
```

---

guess_extension/3

Returns the preferred file extension associated with a MIME type.

Compilation flags:
```
static
```

Template:
```
guess_extension(Type,Extension,Strict)
```
Mode and number of proofs:
```
guess_extension(+atom,-atom,+boolean) - zero_or_one
```

---

guess_all_extensions/2

Returns all known file extensions associated with a MIME type using lenient lookup.

Compilation flags:

```
static
```

Template:

```
guess_all_extensions(Type,Extensions)
```

Mode and number of proofs:

```
guess_all_extensions(+atom,-list(atom)) - one
```

---

guess_all_extensions/3

Returns all known file extensions associated with a MIME type.

Compilation flags:

```
static
```

Template:

```
guess_all_extensions(Type,Extensions,Strict)
```

Mode and number of proofs:

```
guess_all_extensions(+atom,-list(atom),+boolean) - one
```

---

add_type/2

Adds a lenient runtime mapping from a MIME type to a file extension.

Compilation flags:

```
static
```

Template:

```
add_type(Type,Extension)
```

Mode and number of proofs:

```
add_type(+atom,+atom) - one
```

---

add_type/3

Adds a runtime mapping from a MIME type to a file extension.

Compilation flags:
`static`

Template:
`add_type(Type,Extension,Strict)`
Mode and number of proofs:
`add_type(+atom,+atom,+boolean) - one`

---

read_mime_types/2

Reads a `mime.types`-style file returning a list of `Extension-Type` pairs.

Compilation flags:
`static`

Template:
`read_mime_types(File,Pairs)`
Mode and number of proofs:
`read_mime_types(+atom,-list(pair(atom,atom))) - one`

---

load/1

Loads a `mime.types`-style file as a lenient runtime overlay.

Compilation flags:
`static`

Template:
`load(File)`
Mode and number of proofs:
`load(+atom) - one`

---

load/2

Loads a `mime.types`-style file as a strict or lenient runtime overlay.

Compilation flags:
`static`

Template:
`load(File,Strict)`
Mode and number of proofs:
`load(+atom,+boolean)` - `one`

---

reset/0

Clears all runtime MIME type overlays.

Compilation flags:
`static`

Mode and number of proofs:
`reset` - `one`

---

suffix_alias/2

Returns suffix aliases used before splitting encoding and type suffixes.

Compilation flags:
`static`

Template:
`suffix_alias(Alias,Expanded)`
Mode and number of proofs:
`suffix_alias(?atom,?atom)` - `zero_or_more`

---

encoding_suffix/2

Returns content-encoding suffix mappings.

Compilation flags:
`static`

Template:
`encoding_suffix(Extension,Encoding)`

Mode and number of proofs:
`encoding_suffix(?atom,?atom)` - `zero_or_more`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`runtime_type_/3`

Table of runtime MIME type overlays.

Compilation flags:
`dynamic`

Template:
`runtime_type_(Strict,Extension,Type)`
Mode and number of proofs:
`runtime_type_(?boolean,?atom,?atom)` - `zero_or_more`

Operators

(none)

1.221 minizinc

object

1.221.1 `minizinc`

Process-backed MiniZinc CLI solving, source generation, and collected result inspection.

Availability:
`logtalk_load(minizinc(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-09

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *minizinc_protocol*
Imports:

`public` *options*

Uses:

json(ObjectRepresentation,PairRepresentation,StringRepresentation)
list
minizinc_output
minizinc_process
minizinc_program
os
reader
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *model/2* *option/2* *option/3* *print_program/1* *print_result/1* *program/1* *program/2* *result_property/2* *solve/2* *solve/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.221.2 `minizinc_expansion`

Opt-in directive expansion translating MiniZinc declaration directives into canonical statements for pure problem objects.

Availability:

`logtalk_load(minizinc(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-11-10

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
minizinc_program
type

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.221.3 `minizinc_output`

Decoding of the MiniZinc `--json-stream` message sequence into a collected result.

Availability:
`logtalk_load(minizinc(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Uses:
json(ObjectRepresentation,PairRepresentation,StringRepresentation)
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - `decode/5`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`decode/5`

Decodes CLI JSON-stream output and process metadata into a collected `minizinc_result(Status, Solutions, Properties)` term.

Compilation flags:
`static`

Template:
`decode(Codes,RawStatus,NormalizedStatus,Diagnostics,Result)`

Mode and number of proofs:
`decode(+list(code),+term,+compound,+list(code),-compound)` - `one_or_error`

Exceptions:
`Codes` is a partial list or a list with an element `Element` which is a variable:
`instantiation_error`
`Codes` is neither a partial list nor a list of character codes:
`type_error(codes,Codes)`
A `--json-stream` line is not valid JSON:
`domain_error(minizinc_output,json-Line)`
A JSON message field has an invalid type or value:
`domain_error(minizinc_output,Field-Value)`
A JSON message is missing a required field:
`domain_error(minizinc_output,Field-missing)`
`RawStatus` is not a ground term:
`instantiation_error`

`NormalizedStatus` is not a ground term:
`instantiation_error`
`NormalizedStatus` is not `exit(0)` and no JSON-stream message was produced:
`domain_error(minizinc_process_status,NormalizedStatus)`
`Diagnostics` is a partial list or a list with an element `Element` which is a variable:
`instantiation_error`
`Diagnostics` is neither a partial list nor a list of character codes:
`type_error(codes,Diagnostics)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.221.4 `minizinc_problem_protocol`

Protocol for ordered, ground MiniZinc statement providers.

Availability:
`logtalk_load(minizinc(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-09

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

    - *minizinc_statement/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

minizinc_statement/2

Enumerates, by backtracking, canonical MiniZinc statements in source order and their origins.

Compilation flags:
`static`

Template:
`minizinc_statement(Statement,Origin)`
Mode and number of proofs:
`minizinc_statement(-compound,-term)` - `zero_or_more`

Remarks:

- Origin: Either `source(File, Start, End)` or `unknown`.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.221.5 minizinc_process

Isolated operating-system shell transport for the MiniZinc CLI.

Availability:
`logtalk_load(minizinc(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-09

Compilation flags:

`static, context_switching_calls`

Uses:

list
os
process
reader
type
user
utf_8

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *quote/3*
  - *run/8*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

quote/3

Quotes an atom as one POSIX shell or Windows PowerShell literal, without interpolation.

Compilation flags:

`static`

Template:

`quote(Platform,Argument,Quoted)`

Mode and number of proofs:

`quote(+atom,+atom,-atom)` - `one_or_error`

Exceptions:

`Platform` is a variable:

`instantiation_error`

`Argument` is a variable:

`instantiation_error`

`Platform` is neither a variable nor an atom:

`type_error(atom,Platform)`

`Argument` is neither a variable nor an atom:

`type_error(atom,Argument)`

`Platform` is an atom but neither `posix` nor `windows`:

`domain_error(minizinc_shell_platform,Platform)`

`Argument` is an atom but contains a null character:

`domain_error(minizinc_shell_argument,Argument)`

`run/8`

Runs the CLI using an owned directory and separate UTF-8 output files, returning only after cleanup.

Compilation flags:

`static`

Template:

`run(Executable,Arguments,Source,DataPath,Codes,Diagnostics,RawStatus,NormalizedStatus)`

Mode and number of proofs:

`run(+atom,+list(atom),+compound,+atom,-list(integer),-list(integer),-term,-compound)` - `one_or_error`

Exceptions:

`Executable`, `Arguments`, `Source`, or `DataPath` is a variable:

`instantiation_error`

An element of the list `Arguments` is a variable:

`instantiation_error`

`Executable` is neither a variable nor an atom:

`type_error(atom,Executable)`

`Arguments` is neither a variable nor a list:

`type_error(list,Arguments)`

An element of the list `Arguments` is neither a variable nor an atom:

`type_error(atom,Argument)`

`DataPath` is neither `none` nor an atom:

`type_error(atom,DataPath)`

`Source` is neither `file(Path)` nor `codes(Codes)`:

`domain_error(minizinc_process_source,Source)`

An argument contains a null character:

`domain_error(minizinc_shell_argument,Argument)`

A launcher dependency does not exist:

`existence_error(command,Command)`

A source or capture file does not exist:

`existence_error(source_sink,Path)`

A source or capture file cannot be opened:

`permission_error(open,source_sink,Path)`

An owned directory cannot be found during cleanup:

`existence_error(directory,Directory)`

An owned file or directory cannot be removed:

`permission_error(delete,source_sink,Path)`

A character sequence is not valid UTF-8:

`representation_error(utf_8)`

A launcher operation fails:

```
minizinc_transport(Stage,Status,Details)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.221.6 `minizinc_program`

Canonical MiniZinc statement validation and source generation.

Availability:

```
logtalk_load(minizinc(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-15

Compilation flags:

```
static, context_switching_calls
```

Uses:

list
type

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *serialize/2*
  - *serialize/4*
  - *valid_identifier/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

serialize/2

Serializes ground canonical MiniZinc statements to character codes.

Compilation flags:
`static`

Template:
`serialize(Statements,Codes)`
Mode and number of proofs:
`serialize(+list(compound),-list(integer))` - `one_or_error`

Exceptions:
`Statements` is a variable:
`instantiation_error`
An element `Statement` of the list `Statements` is a variable:
`instantiation_error`
`Statements` is neither a variable nor a list:
`type_error(list,Statements)`
An element `Statement` of the list `Statements` is not a valid statement:
`domain_error(minizinc_statement,Statement)`

---

serialize/4

Serializes statements with origins and returns `line(Start, End, Origin)` source-map entries.

Compilation flags:
`static`

Template:
`serialize(Statements,Origins,Codes,SourceMap)`
Mode and number of proofs:
`serialize(+list(compound),+list,-list(integer),-list(compound))` - `one_or_error`

Exceptions:
`Statements` or `Origins` is a variable:
`instantiation_error`
An element `Statement` of the list `Statements` is a variable:
`instantiation_error`
An element `Origin` of the list `Origins` is a variable:
`instantiation_error`
`Statements` is neither a variable nor a list:
`type_error(list,Statements)`
`Origins` is neither a variable nor a list:
`type_error(list,Origins)`
An element `Statement` of the list `Statements` is not a valid statement:

`domain_error(minizinc_statement,Statement)`
`Origins` has an element that is neither `unknown` nor a valid `source(File, Start, End)` term, or its length differs from `Statements`:
`domain_error(minizinc_origins,Origins)`

`valid_identifier/1`

True if the argument is a supported, non-reserved MiniZinc identifier atom.

Compilation flags:
`static`

Template:
`valid_identifier(Name)`
Mode and number of proofs:
`valid_identifier(@term)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.221.7 `minizinc_protocol`

Protocol for process-backed MiniZinc solving, source generation, and result inspection.

Availability:
`logtalk_load(minizinc(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *solve/2*
  - *solve/3*
  - *program/1*
  - *program/2*
  - *print_program/1*
  - *model/2*
  - *result_property/2*
  - *print_result/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

solve/2

Solves a source with default options using the external MiniZinc CLI and returns a collected result after cleanup. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`.

Compilation flags:
`static`

Template:
`solve(Source,Result)`

Mode and number of proofs:
`solve(+compound,-compound)` - `one_or_error`

Exceptions:
`Source` is not a ground term:
`instantiation_error`
`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `minizinc_problem_protocol`:
`domain_error(minizinc_source,Source)`
`Statements` is ground but not a list:
`type_error(list,Statements)`
A statement enumerated by the provider object is not a valid statement:
`domain_error(minizinc_statement,Statement)`
An origin enumerated by the provider object is invalid, or its origins and statements differ in length:
`domain_error(minizinc_origins,Origins)`
`Codes` is ground but not a list of character codes:
`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:
`type_error(integer,Code)`
An element `Code` of the list `Codes` is an integer but not a valid character code:
`domain_error(character_code,Code)`
`Path` is ground but not an atom:
`type_error(atom,Path)`
A launcher argument contains a null character:
`domain_error(minizinc_shell_argument,Argument)`
The `minizinc` executable or a shell dependency does not exist:
`existence_error(command,Command)`
An owned directory cannot be found during cleanup:
`existence_error(directory,Directory)`
An owned file or directory cannot be removed:
`permission_error(delete,source_sink,Path)`
A character sequence is not valid UTF-8:
`representation_error(utf_8)`
A launcher operation fails:
`minizinc_transport(Stage,Status,Details)`
Solver output is invalid or inconsistent with its exit status:
`minizinc_solver_error(Formal,Status,Diagnostics,SourceMap)`

`solve/3`

Solves a source using solver selection, executable, enumeration, data, and resource options. Repeated options use the first occurrence. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`.

Compilation flags:
`static`

Template:
`solve(Source,Result,Options)`
Mode and number of proofs:
`solve(+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Source` is not a ground term:
`instantiation_error`
`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `minizinc_problem_protocol`:
`domain_error(minizinc_source,Source)`
`Statements` is ground but not a list:
`type_error(list,Statements)`
A statement enumerated by the provider object is not a valid statement:
`domain_error(minizinc_statement,Statement)`
`Codes` is ground but not a list of character codes:
`type_error(codes,Codes)`
An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(minizinc_origins,Origins)`

A launcher argument contains a null character:

`domain_error(minizinc_shell_argument,Argument)`

The `minizinc` executable or a shell dependency does not exist:

`existence_error(command,Command)`

An owned directory cannot be found during cleanup:

`existence_error(directory,Directory)`

An owned file or directory cannot be removed:

`permission_error(delete,source_sink,Path)`

A character sequence is not valid UTF-8:

`representation_error(utf_8)`

A launcher operation fails:

`minizinc_transport(Stage,Status,Details)`

Solver output is invalid or inconsistent with its exit status:

`minizinc_solver_error(Formal,Status,Diagnostics,SourceMap)`

`program/1`

Generates and prints MiniZinc source to the current output, equivalent to `program/2` followed by `print_program/1`, without invoking MiniZinc.

Compilation flags:

`static`

Template:

`program(Source)`

Mode and number of proofs:

`program(+compound)` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `minizinc_problem_protocol`:

`domain_error(minizinc_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(minizinc_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(minizinc_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

The current output stream does not allow writing:

`permission_error(output,stream,Stream)`

---

program/2

Returns MiniZinc source codes from `object(Object)`, `terms(Statements)`, `codes(Codes)`, or `file(Path)` without invoking MiniZinc.

Compilation flags:

`static`

Template:

`program(Source,Codes)`

Mode and number of proofs:

`program(+compound,-list(integer))` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `minizinc_problem_protocol`:

`domain_error(minizinc_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(minizinc_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(minizinc_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

`print_program/1`

Prints MiniZinc character codes to the current output without adding a newline or invoking MiniZinc.

Compilation flags:

`static`

Template:

`print_program(Codes)`

Mode and number of proofs:

`print_program(+list(integer))` - `one_or_error`

Exceptions:

`Codes` is not a ground term:

`instantiation_error`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

The current output stream does not allow writing:

`permission_error(output,stream,Stream)`

---

model/2

Enumerates, by backtracking, collected `solution(Values, Objective, Time)` records without starting a process.

Compilation flags:
`static`

Template:
`model(Result,Solution)`
Mode and number of proofs:
`model(+compound,-compound)` - `zero_or_more`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected MiniZinc result:
`domain_error(minizinc_result,Result)`

---

result_property/2

Enumerates, by backtracking, `status(Status)` and the stored result properties.

Compilation flags:
`static`

Template:
`result_property(Result,Property)`
Mode and number of proofs:
`result_property(+compound,?compound)` - `zero_or_more`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected MiniZinc result:
`domain_error(minizinc_result,Result)`

---

print_result/1

Prints collected solutions, objective values, and solver status to the current output using MiniZinc-style markers, without invoking the solver.

Compilation flags:
`static`

Template:

```
print_result(Result)
```

Mode and number of proofs:

```
print_result(+compound) - one_or_error
```

Exceptions:

`Result` is not a ground term:

```
instantiation_error
```

`Result` is not a valid collected MiniZinc result:

```
domain_error(minizinc_result,Result)
```

The current output stream does not allow writing:

```
permission_error(output,stream,Stream)
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.222 mlp_classifier

object

1.222.1 `mlp_classifier`

Multi-layer perceptron classifier with configurable hidden layers and activation function, softmax output, and backpropagation training.

Availability:

```
logtalk_load(mlp_classifier(loader))
```

Author: Paulo Moura
Version: 2:0:0
Date: 2026-10-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *probabilistic_classifier_common*

Uses:

fast_random(Algorithm)
format

linear_algebra
list
type

Remarks:
(none)

Inherited public predicates:
check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *logistic_regression_classifier*, *sgd_classifier*

1.223 modified_z_score_anomaly_detector

object

1.223.1 modified_z_score_anomaly_detector

Statistical modified Z-score anomaly detector for continuous datasets. Learns per-attribute sample median and median absolute deviation from baseline training examples selected from a dataset object implementing the `anomaly_dataset_protocol` protocol and returns a detector term that can be used for scoring, prediction, and export.

Availability:
`logtalk_load(modified_z_score_anomaly_detector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *anomaly_detector_common*

Uses:
format
list
numberlist
pairs
sample
type

Remarks:
(none)

Inherited public predicates:
anomaly_detector_options/2 *check_anomaly_detector/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict/4* *print_anomaly_detector/1* *score/3* *score_all/3* *valid_anomaly_detector/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

anomaly_dataset_protocol, *anomaly_detector_protocol*, *z_score_anomaly_detector*, *isolation_forest_anomaly_detector*, *knn_distance_anomaly_detector*, *lof_anomaly_detector*

1.224 mqtt

object

1.224.1 `mqtt`

Portable MQTT 5 client predicates using `http_transport_protocol` implementations.

Availability:
`logtalk_load(mqtt(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-10-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
`public` *http_origin_site_helpers*

Uses:
byte_order
list
type
url(Representation)
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *connect/3*
  - *connect/4*
  - *disconnect/2*
  - *connection_alive/1*
  - *packet_type/2*
  - *reason_code/3*
  - *publish/4*
  - *subscribe/4*
  - *unsubscribe/4*
  - *receive/3*
  - *send_pingreq/1*
  - *ping/2*
  - *encode_packet/2*
  - *decode_packet/2*
- *Protected predicates*
- *Private predicates*
  - *resolve_address/4*
  - *resolve_transport/3*
  - *append_tls_transport/3*
  - *write_packet/2*
  - *read_packet/2*
  - *encode_uint8/2*
  - *decode_uint8/3*
  - *encode_uint16/2*
  - *decode_uint16/3*
  - *encode_uint32/2*
  - *decode_uint32/3*
  - *encode_varint/2*
  - *decode_varint/3*
  - *encode_utf8_string/2*
  - *decode_utf8_string/3*
  - *encode_binary_data/2*
  - *decode_binary_data/3*
  - *encode_utf8_string_pair/2*
  - *decode_utf8_string_pair/3*
  - *encode_properties/2*
  - *decode_properties/3*
  - *encode_property_list/2*
  - *decode_property_list/2*
  - *encode_property/2*
  - *decode_property/3*
  - *mqtt_property/3*
  - *encode_fixed_header/4*
  - *decode_fixed_header/5*
- *Operators*

Public predicates

connect/3

Opens a transport connection for an MQTT or MQTT-over-TLS address and completes the MQTT CONNECT/CONNACK exchange.

Compilation flags:

`static`

Template:

`connect(Address,Connection,Options)`

Mode and number of proofs:

`connect(+atom,--compound,+list)` - `one_or_error`

Exceptions:

`Address` is a variable:

`instantiation_error`

`Address` is neither a variable nor an atom:

`type_error(atom,Address)`

`Address` is an atom but not a valid MQTT address:

`domain_error(mqtt_address,Address)`

The server rejects the MQTT connection:

`domain_error(mqtt_connack_reason_code,ReasonCode)`

The server response is not a CONNACK packet:

`domain_error(mqtt_packet,expected(connack,Packet))`

---

connect/4

Opens a transport connection to the given host and port and completes the MQTT CONNECT/CONNACK exchange. The `scheme/1` option selects MQTT over TCP or TLS.

Compilation flags:

`static`

Template:

`connect(Host,Port,Connection,Options)`

Mode and number of proofs:

`connect(+atom,+integer,--compound,+list)` - `one_or_error`

Exceptions:

`Host` or `Port` is a variable:

`instantiation_error`

`Host` is neither a variable nor an atom:

`type_error(atom,Host)`

`Port` is neither a variable nor an integer:

`type_error(integer,Port)`

`Port` is a negative integer:

`domain_error(non_negative_integer,Port)`

The server rejects the MQTT connection:

`domain_error(mqtt_connack_reason_code,ReasonCode)`

The server response is not a CONNACK packet:

`domain_error(mqtt_packet,expected(connack,Packet))`

---

disconnect/2

Closes an MQTT transport connection. MQTT DISCONNECT packet exchange will be layered on this transport operation.

Compilation flags:

`static`

Template:

`disconnect(Connection,Options)`

Mode and number of proofs:

`disconnect(+compound,+list)` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is not a valid option:

`domain_error(option,Option)`

---

connection_alive/1

True if the term is an MQTT connection handle created by this object.

Compilation flags:

`static`

Template:

`connection_alive(Connection)`

Mode and number of proofs:

`connection_alive(+compound)` - `zero_or_one`

---

packet_type/2

Table of MQTT control packet type names and numeric codes.

Compilation flags:
`static`

Template:
`packet_type(Type,Code)`
Mode and number of proofs:
`packet_type(?atom,?integer)` - `zero_or_more`

---

reason_code/3

Table of MQTT reason code names and numeric codes by packet type.

Compilation flags:
`static`

Template:
`reason_code(PacketType,Name,Code)`
Mode and number of proofs:
`reason_code(?atom,?atom,?integer)` - `zero_or_more`

---

publish/4

Publishes a message and waits synchronously for the required MQTT acknowledgement.

Compilation flags:
`static`

Template:
`publish(Connection,Topic,Payload,Options)`
Mode and number of proofs:
`publish(+compound,+atom,+term,+list)` - `one_or_error`

Exceptions:
`Topic` is not a valid MQTT topic name:
`domain_error(mqtt_topic_name,Topic)`
`Options` contains an invalid packet identifier:
`domain_error(mqtt_packet_identifier,PacketIdentifier)`
`Options` contains an invalid QoS value:
`domain_error(mqtt_qos,QoS)`
The server response is not the expected acknowledgement packet:
`domain_error(mqtt_packet,expected(Type-PacketIdentifier,Packet))`

---

subscribe/4

Subscribes to topic filters and waits synchronously for the matching SUBACK.

Compilation flags:
`static`

Template:
`subscribe(Connection,TopicFilters,Result,Options)`
Mode and number of proofs:
`subscribe(+compound,+list,--list,+list)` - `one_or_error`

Exceptions:
`TopicFilters` is empty:
`domain_error(mqtt_subscriptions,[])`
An element `TopicFilter` is not a valid MQTT topic filter:
`domain_error(mqtt_topic_filter,TopicFilter)`
`Options` contains invalid subscription options:
`domain_error(mqtt_subscription_options,Byte)`
The server response is not the expected SUBACK packet:
`domain_error(mqtt_packet,expected(suback-PacketIdentifier,Packet))`

---

unsubscribe/4

Unsubscribes from topic filters and waits synchronously for the matching UNSUBACK.

Compilation flags:
`static`

Template:
`unsubscribe(Connection,TopicFilters,Result,Options)`
Mode and number of proofs:
`unsubscribe(+compound,+list,--list,+list)` - `one_or_error`

Exceptions:
`TopicFilters` is empty:
`domain_error(mqtt_topic_filters,[])`
An element `TopicFilter` is not a valid MQTT topic filter:
`domain_error(mqtt_topic_filter,TopicFilter)`
The server response is not the expected UNSUBACK packet:
`domain_error(mqtt_packet,expected(unsuback-PacketIdentifier,Packet))`

---

receive/3

Receives the next MQTT packet.

Compilation flags:
`static`

Template:
`receive(Connection,Message,Options)`
Mode and number of proofs:
`receive(+compound,--compound,+list)` - `one_or_error`

Exceptions:
The connection input ends before a complete packet is read:
`domain_error(mqtt_packet_stream,unexpected_end_of_file)`
The received bytes do not encode a valid MQTT packet:
`domain_error(mqtt_packet,Packet)`

---

send_pingreq/1

Sends a PINGREQ packet.

Compilation flags:
`static`

Template:
`send_pingreq(Connection)`
Mode and number of proofs:
`send_pingreq(+compound)` - `one_or_error`

Exceptions:
The PINGREQ packet cannot be written to the connection output stream:
`resource_error(mqtt_packet_stream)`

---

ping/2

Sends PINGREQ and waits synchronously for PINGRESP.

Compilation flags:
`static`

Template:
`ping(Connection,Options)`
Mode and number of proofs:
`ping(+compound,+list)` - `one_or_error`

Exceptions:

The connection input ends before a complete packet is read:

```
domain_error(mqtt_packet_stream,unexpected_end_of_file)
```

The server response is not a PINGRESP packet:

```
domain_error(mqtt_packet,expected(pingresp,Packet))
```

---

encode_packet/2

Encodes a normalized MQTT packet term.

Compilation flags:

```
static
```

Template:

```
encode_packet(Packet,Bytes)
```

Mode and number of proofs:

```
encode_packet(+compound,--list(integer)) - one_or_error
```

Exceptions:

`Packet` contains an invalid MQTT field value:

```
domain_error(mqtt_packet,Packet)
```

`Packet` has a packet type that is not yet supported:

```
resource_error(mqtt_packet_support)
```

---

decode_packet/2

Decodes bytes into a normalized MQTT packet term.

Compilation flags:

```
static
```

Template:

```
decode_packet(Bytes,Packet)
```

Mode and number of proofs:

```
decode_packet(+list(integer),--compound) - one_or_error
```

Exceptions:

`Bytes` contains an invalid MQTT packet encoding:

```
domain_error(mqtt_packet,Bytes)
```

`Bytes` contains bytes after the encoded packet:

```
domain_error(mqtt_packet_trailing_bytes,Rest)
```

`Bytes` encodes a packet type that is not yet supported:

```
resource_error(mqtt_packet_support)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

resolve_address/4

Resolves an MQTT address into scheme, host, and port.

Compilation flags:
`static`

Template:
`resolve_address(Address,Scheme,Host,Port)`
Mode and number of proofs:
`resolve_address(+atom,-atom,-atom,-integer)` - `one_or_error`

Exceptions:
`Address` is a variable:
`instantiation_error`
`Address` is neither a variable nor an atom:
`type_error(atom,Address)`
`Address` is an atom but not a valid MQTT address:
`domain_error(mqtt_address,Address)`
`Address` contains an invalid authority:
`domain_error(mqtt_address_authority,Authority)`

---

resolve_transport/3

Resolves an explicit or default transport object for the given MQTT scheme.

Compilation flags:
`static`

Template:
`resolve_transport(Scheme,Transport0,Transport)`
Mode and number of proofs:
`resolve_transport(+atom,+object_identifier,-object_identifier)` - `one_or_error`

Exceptions:
`Transport0` is a variable:
`instantiation_error`
`Transport0` is neither a variable nor an existing object:
`existence_error(object,Transport0)`
`Transport0` does not implement `http_transport_protocol`:
`domain_error(http_transport_protocol_object,Transport0)`
`Transport0` does not support `Scheme`:

```
consistency_error(mqtt_options,scheme(Scheme),transport(Transport0))
```

---

append_tls_transport/3

Adds `connection_transport(tls)` to connection options for MQTT-over-TLS unless already specified.

Compilation flags:
`static`

Template:
`append_tls_transport(Scheme,Options,OptionsWithTransport)`
Mode and number of proofs:
`append_tls_transport(+atom,+list,-list)` - `one`

---

write_packet/2

Encodes and writes an MQTT packet to a binary output stream.

Compilation flags:
`static`

Template:
`write_packet(Output,Packet)`
Mode and number of proofs:
`write_packet(+stream,+compound)` - `one_or_error`

Exceptions:
`Packet` is not a supported normalized MQTT packet:
`resource_error(mqtt_packet_support)`
`Output` cannot be written or flushed:
`resource_error(mqtt_packet_stream)`

---

read_packet/2

Reads and decodes one MQTT packet from a binary input stream.

Compilation flags:
`static`

Template:
`read_packet(Input,Packet)`
Mode and number of proofs:

`read_packet(+stream,--compound)` - `one_or_error`

Exceptions:

`Input` ends before a complete packet is read:

`domain_error(mqtt_packet_stream,unexpected_end_of_file)`

`Input` contains an invalid MQTT packet encoding:

`domain_error(mqtt_packet,Packet)`

---

encode_uint8/2

Encodes an unsigned 8-bit integer.

Compilation flags:

`static`

Template:

`encode_uint8(Value,Bytes)`

Mode and number of proofs:

`encode_uint8(+integer,-list(integer))` - `one_or_error`

Exceptions:

`Value` is a variable:

`instantiation_error`

`Value` is neither a variable nor an integer:

`type_error(integer,Value)`

`Value` is outside the unsigned 8-bit integer range:

`domain_error(between(0,255),Value)`

---

decode_uint8/3

Decodes an unsigned 8-bit integer.

Compilation flags:

`static`

Template:

`decode_uint8(Bytes,Value,Rest)`

Mode and number of proofs:

`decode_uint8(+list(integer),-integer,-list(integer))` - `one_or_error`

Exceptions:

The first element of `Bytes` is not a byte:

`type_error(byte,Byte)`

---

encode_uint16/2

Encodes an unsigned 16-bit integer in network byte order.

Compilation flags:
`static`

Template:
`encode_uint16(Value,Bytes)`
Mode and number of proofs:
`encode_uint16(+integer,-list(integer))` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor an integer:
`type_error(integer,Value)`
`Value` is outside the unsigned 16-bit integer range:
`domain_error(between(0,65535),Value)`

---

decode_uint16/3

Decodes an unsigned 16-bit integer in network byte order.

Compilation flags:
`static`

Template:
`decode_uint16(Bytes,Value,Rest)`
Mode and number of proofs:
`decode_uint16(+list(integer),-integer,-list(integer))` - `one_or_error`

Exceptions:
One of the first two elements of `Bytes` is not a byte:
`type_error(byte,Byte)`

---

encode_uint32/2

Encodes an unsigned 32-bit integer in network byte order.

Compilation flags:
`static`

Template:
`encode_uint32(Value,Bytes)`

Mode and number of proofs:

`encode_uint32(+integer,-list(integer))` - `one_or_error`

Exceptions:

`Value` is a variable:

`instantiation_error`

`Value` is neither a variable nor an integer:

`type_error(integer,Value)`

`Value` is outside the unsigned 32-bit integer range:

`domain_error(between(0,4294967295),Value)`

---

decode_uint32/3

Decodes an unsigned 32-bit integer in network byte order.

Compilation flags:

`static`

Template:

`decode_uint32(Bytes,Value,Rest)`

Mode and number of proofs:

`decode_uint32(+list(integer),-integer,-list(integer))` - `one_or_error`

Exceptions:

One of the first four elements of `Bytes` is not a byte:

`type_error(byte,Byte)`

---

encode_varint/2

Encodes an MQTT variable byte integer.

Compilation flags:

`static`

Template:

`encode_varint(Value,Bytes)`

Mode and number of proofs:

`encode_varint(+integer,-list(integer))` - `one_or_error`

Exceptions:

`Value` is a variable:

`instantiation_error`

`Value` is neither a variable nor an integer:

`type_error(integer,Value)`

`Value` is outside the MQTT variable byte integer range:
`domain_error(between(0,268435455),Value)`

`decode_varint/3`

Decodes an MQTT variable byte integer, rejecting non-minimal encodings and overflows.

Compilation flags:
`static`

Template:
`decode_varint(Bytes,Value,Rest)`
Mode and number of proofs:
`decode_varint(+list(integer),-integer,-list(integer))` - `one_or_error`

Exceptions:
`Bytes` does not start with a valid, minimally encoded MQTT variable byte integer:
`domain_error(mqtt_variable_byte_integer,Consumed)`

`encode_utf8_string/2`

Encodes an MQTT UTF-8 encoded string with a two-byte length prefix.

Compilation flags:
`static`

Template:
`encode_utf8_string(String,Bytes)`
Mode and number of proofs:
`encode_utf8_string(+atom,-list(integer))` - `one_or_error`

Exceptions:
`String` is a variable:
`instantiation_error`
`String` is neither a variable nor atom:
`type_error(atom,String)`
`String` is not a valid MQTT UTF-8 string:
`domain_error(mqtt_utf8_string,Codes)`

---

decode_utf8_string/3

Decodes an MQTT UTF-8 encoded string with a two-byte length prefix.

Compilation flags:
`static`

Template:
`decode_utf8_string(Bytes,String,Rest)`
Mode and number of proofs:
`decode_utf8_string(+list(integer),-atom,-list(integer))` - `one_or_error`

Exceptions:
`Bytes` does not start with a valid MQTT UTF-8 string:
`domain_error(mqtt_utf8_string,Codes)`

---

encode_binary_data/2

Encodes MQTT binary data with a two-byte length prefix.

Compilation flags:
`static`

Template:
`encode_binary_data(Data,Bytes)`
Mode and number of proofs:
`encode_binary_data(+list(integer),-list(integer))` - `one_or_error`

Exceptions:
`Data` is not a list of bytes:
`type_error(list(byte),Data)`
`Data` is longer than 65535 bytes:
`domain_error(between(0,65535),Length)`

---

decode_binary_data/3

Decodes MQTT binary data with a two-byte length prefix.

Compilation flags:
`static`

Template:
`decode_binary_data(Bytes,Data,Rest)`
Mode and number of proofs:
`decode_binary_data(+list(integer),-list(integer),-list(integer))` - `one_or_error`

Exceptions:

The first two elements of `Bytes` do not encode a valid length:

`type_error(byte,Byte)`

---

encode_utf8_string_pair/2

Encodes an MQTT UTF-8 string pair as two MQTT UTF-8 encoded strings.

Compilation flags:

`static`

Template:

`encode_utf8_string_pair(Pair,Bytes)`

Mode and number of proofs:

`encode_utf8_string_pair(+compound,-list(integer))` - `one_or_error`

Exceptions:

An element of `Pair` is not an atom:

`type_error(atom,String)`

An element of `Pair` is not a valid MQTT UTF-8 string:

`domain_error(mqtt_utf8_string,Codes)`

---

decode_utf8_string_pair/3

Decodes an MQTT UTF-8 string pair from two MQTT UTF-8 encoded strings.

Compilation flags:

`static`

Template:

`decode_utf8_string_pair(Bytes,Pair,Rest)`

Mode and number of proofs:

`decode_utf8_string_pair(+list(integer),-compound,-list(integer))` - `one_or_error`

Exceptions:

`Bytes` does not start with two valid MQTT UTF-8 strings:

`domain_error(mqtt_utf8_string,Codes)`

---

encode_properties/2

Encodes an MQTT property list with its variable byte integer length prefix.

Compilation flags:
`static`

Template:
`encode_properties(Properties,Bytes)`
Mode and number of proofs:
`encode_properties(+list,-list(integer))` - `one_or_error`

Exceptions:
`Properties` is a variable or a partial list:
`instantiation_error`
`Properties` is neither a variable nor a list:
`type_error(list,Properties)`
An element `Property` is not a supported MQTT property:
`domain_error(mqtt_property,Property)`

---

decode_properties/3

Decodes an MQTT property list and returns the remaining packet bytes.

Compilation flags:
`static`

Template:
`decode_properties(Bytes,Properties,Rest)`
Mode and number of proofs:
`decode_properties(+list(integer),-list,-list(integer))` - `one_or_error`

Exceptions:
`Bytes` does not contain the declared property bytes:
`domain_error(mqtt_properties,Bytes)`
`Bytes` contains an unknown property identifier:
`domain_error(mqtt_property_identifier,Identifier)`

---

encode_property_list/2

Encodes MQTT properties without the property length prefix.

Compilation flags:
`static`

Template:
`encode_property_list(Properties,Bytes)`
Mode and number of proofs:
`encode_property_list(+list,-list(integer))` - `one_or_error`

Exceptions:
An element `Property` is not a supported MQTT property:
`domain_error(mqtt_property,Property)`

---

decode_property_list/2

Decodes MQTT properties from a property body byte list.

Compilation flags:
`static`

Template:
`decode_property_list(Bytes,Properties)`
Mode and number of proofs:
`decode_property_list(+list(integer),-list)` - `one_or_error`

Exceptions:
`Bytes` contains an unknown property identifier:
`domain_error(mqtt_property_identifier,Identifier)`

---

encode_property/2

Encodes a single MQTT property.

Compilation flags:
`static`

Template:
`encode_property(Property,Bytes)`
Mode and number of proofs:
`encode_property(+compound,-list(integer))` - `one_or_error`

Exceptions:

`Property` is not a supported MQTT property:
`domain_error(mqtt_property,Property)`
`Property` contains an invalid value:
`domain_error(mqtt_property_value,Name-Value)`

---

decode_property/3

Decodes a single MQTT property.

Compilation flags:
`static`

Template:
`decode_property(Bytes,Property,Rest)`
Mode and number of proofs:
`decode_property(+list(integer),-compound,-list(integer))` - `one_or_error`

Exceptions:
`Bytes` starts with an unknown property identifier:
`domain_error(mqtt_property_identifier,Identifier)`
`Bytes` contains an invalid property value:
`domain_error(mqtt_property_value,Name-Value)`

---

mqtt_property/3

Table of MQTT property names, identifiers, and value types.

Compilation flags:
`static`

Template:
`mqtt_property(Name,Identifier,Type)`
Mode and number of proofs:
`mqtt_property(?atom,?integer,?atom)` - `zero_or_more`

---

encode_fixed_header/4

Encodes an MQTT fixed header from packet type, flags, and remaining length.

Compilation flags:
`static`

Template:
`encode_fixed_header(Type,Flags,RemainingLength,Bytes)`
Mode and number of proofs:
`encode_fixed_header(+atom,+integer,+integer,-list(integer))` - `one_or_error`

Exceptions:
`Type` is not a supported MQTT packet type:
`domain_error(mqtt_packet_type,Type)`
`Flags` is invalid for `Type`:
`domain_error(mqtt_fixed_header_flags,Type-Flags)`
`RemainingLength` is outside the MQTT variable byte integer range:
`domain_error(between(0,268435455),RemainingLength)`

---

decode_fixed_header/5

Decodes an MQTT fixed header into packet type, flags, remaining length, and remaining bytes.

Compilation flags:
`static`

Template:
`decode_fixed_header(Bytes,Type,Flags,RemainingLength,Rest)`
Mode and number of proofs:
`decode_fixed_header(+list(integer),-atom,-integer,-integer,-list(integer))` - `one_or_error`

Exceptions:
`Bytes` starts with an unsupported MQTT packet type:
`domain_error(mqtt_packet_type,Code)`
`Bytes` contains invalid fixed header flags:
`domain_error(mqtt_fixed_header_flags,Type-Flags)`
`Bytes` contains an invalid MQTT variable byte integer:
`domain_error(mqtt_variable_byte_integer,Consumed)`

Operators

(none)

1.225 mrmr_feature_selector

object

1.225.1 `mrmr_feature_selector`

Minimum redundancy maximum relevance selection using raw mutual information in bits and the MID difference criterion. Learning requires at least two joint complete observations; otherwise it throws `domain_error(mrmr_usable_examples,Count)`.

Availability:
`logtalk_load(mrmr_feature_selector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *feature_selector_common*
`public` *feature_redundancy*

Uses:
avltree
list
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_selector/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *feature_scores/2* *learn/2* *learn/3* *option/2* *option/3* *print_selector/1* *selected_features/2* *selector_options/2* *valid_option/1* *valid_options/1* *valid_selector/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *check_usable_examples/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`check_usable_examples/1`

Requires at least two jointly complete observations.

Compilation flags:
`static`

Template:
`check_usable_examples(Count)`
Mode and number of proofs:
`check_usable_examples(+non_negative_integer)` - `one_or_error`

Exceptions:
Fewer than two jointly complete observations remain:
`domain_error(mrmr_usable_examples,Count)`

Operators

(none)

1.226 multisets

object

1.226.1 `multisets`

Implementation of multiset operations with repetition over lists.

Availability:
`logtalk_load(multisets(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-05-21

Compilation flags:
`static, context_switching_calls`

Implements:

`public` *multisets_protocol*

Uses:

fast_random(Algorithm)
list
natural

Remarks:

(none)

Inherited public predicates:

count_distinct_multisets/3 *count_multisets/3* *distinct_multiset/3* *distinct_multiset/4* *distinct_multiset_index/4* *distinct_multisets/3* *distinct_multisets/4* *multiset/3* *multiset/4* *multiset_index/4* *multiset_index/5* *multisets/3* *multisets/4* *next_multiset/3* *nth_distinct_multiset/4* *nth_multiset/4* *nth_multiset/5* *previous_multiset/3* *random_distinct_multiset/3* *random_multiset/3* *sample_distinct_multisets/4* *sample_multisets/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.226.2 `multisets_protocol`

Protocol for multiset operations over lists.

Availability:

`logtalk_load(multisets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-12

Compilation flags:

`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *multisets/3*
  - *multiset/3*
  - *multisets/4*
  - *multiset/4*
  - *distinct_multisets/3*
  - *distinct_multiset/3*
  - *distinct_multisets/4*
  - *distinct_multiset/4*
  - *nth_multiset/4*
  - *nth_multiset/5*
  - *multiset_index/4*
  - *multiset_index/5*
  - *count_multisets/3*
  - *count_distinct_multisets/3*
  - *nth_distinct_multiset/4*
  - *distinct_multiset_index/4*
  - *random_multiset/3*
  - *sample_multisets/4*
  - *random_distinct_multiset/3*
  - *sample_distinct_multisets/4*
  - *next_multiset/3*
  - *previous_multiset/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

multisets/3

Generates all K-multisets (unordered K-element selections with replacement) of a list using default order.

Compilation flags:
`static`

Template:
`multisets(K,List,Multisets)`
Mode and number of proofs:
`multisets(+integer,+list,-list)` - `one`

---

multiset/3

True iff the third argument is a K-multiset with replacement of a list using default order.

Compilation flags:
`static`

Template:
`multiset(K,List,Multiset)`
Mode and number of proofs:
`multiset(+integer,+list,-list)` - `one_or_more`

---

multisets/4

Generates all K-multisets with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`multisets(K,List,Order,Multisets)`
Mode and number of proofs:
`multisets(+integer,+list,+atom,-list)` - `one`

---

multiset/4

True iff the fourth argument is a K-multiset with replacement with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`multiset(K,List,Order,Multiset)`
Mode and number of proofs:
`multiset(+integer,+list,+atom,-list)` - `one_or_more`

---

distinct_multisets/3

Generates all distinct K-multisets (deduplicating repeated values in the input list) using default order.

Compilation flags:
`static`

Template:
`distinct_multisets(K,List,Multisets)`
Mode and number of proofs:
`distinct_multisets(+integer,+list,-list)` - `one`

---

distinct_multiset/3

True iff the third argument is a distinct K-multiset with replacement of a list using default order.

Compilation flags:
`static`

Template:
`distinct_multiset(K,List,Multiset)`
Mode and number of proofs:
`distinct_multiset(+integer,+list,-list)` - `one_or_more`

---

distinct_multisets/4

Generates all distinct K-multisets with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_multisets(K,List,Order,Multisets)`
Mode and number of proofs:
`distinct_multisets(+integer,+list,+atom,-list)` - `one`

---

distinct_multiset/4

True iff the fourth argument is a distinct K-multiset with replacement with the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_multiset(K,List,Order,Multiset)`
Mode and number of proofs:
`distinct_multiset(+integer,+list,+atom,-list)` - `one_or_more`

---

nth_multiset/4

Returns the K-multiset at a given zero-based index using default order.

Compilation flags:
`static`

Template:
`nth_multiset(K,List,Index,Multiset)`
Mode and number of proofs:
`nth_multiset(+integer,+list,+integer,-list)` - `zero_or_one`

---

nth_multiset/5

Returns the K-multiset at a given zero-based index in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`nth_multiset(K,List,Order,Index,Multiset)`
Mode and number of proofs:
`nth_multiset(+integer,+list,+atom,+integer,-list)` - `zero_or_one`

---

multiset_index/4

Returns the zero-based index of a K-multiset using default order.

Compilation flags:
`static`

Template:
`multiset_index(K,List,Multiset,Index)`
Mode and number of proofs:
`multiset_index(+integer,+list,+list,-integer)` - `zero_or_one`

---

multiset_index/5

Returns the zero-based index of a K-multiset in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`multiset_index(K,List,Order,Multiset,Index)`
Mode and number of proofs:
`multiset_index(+integer,+list,+atom,+list,-integer)` - `zero_or_one`

---

count_multisets/3

Counts the number of K-multisets with replacement of a list.

Compilation flags:
```
static
```

Template:
```
count_multisets(K,List,Count)
```
Mode and number of proofs:
```
count_multisets(+integer,+list,-integer) - one
```

---

count_distinct_multisets/3

Counts the number of distinct K-multisets with replacement of a list (deduplicating repeated values in the input list).

Compilation flags:
```
static
```

Template:
```
count_distinct_multisets(K,List,Count)
```
Mode and number of proofs:
```
count_distinct_multisets(+integer,+list,-integer) - one
```

---

nth_distinct_multiset/4

Returns the distinct K-multiset at a given zero-based index in default generation order.

Compilation flags:
```
static
```

Template:
```
nth_distinct_multiset(K,List,Index,Multiset)
```
Mode and number of proofs:
```
nth_distinct_multiset(+integer,+list,+integer,-list) - zero_or_one
```

---

distinct_multiset_index/4

Returns the zero-based index of a distinct K-multiset in default generation order.

Compilation flags:
```
static
```

Template:
```
distinct_multiset_index(K,List,Multiset,Index)
```
Mode and number of proofs:
```
distinct_multiset_index(+integer,+list,+list,-integer) - zero_or_one
```

---

random_multiset/3

Returns a random K-multiset with replacement of a list.

Compilation flags:
```
static
```

Template:
```
random_multiset(K,List,Multiset)
```
Mode and number of proofs:
```
random_multiset(+integer,+list,-list) - zero_or_one
```

---

sample_multisets/4

Returns SampleCount random K-multisets with replacement of a list, sampled with replacement.

Compilation flags:
```
static
```

Template:
```
sample_multisets(K,List,SampleCount,Samples)
```
Mode and number of proofs:
```
sample_multisets(+integer,+list,+integer,-list) - zero_or_one
```

---

random_distinct_multiset/3

Returns a random distinct K-multiset with replacement of a list (deduplicating repeated values in the input list).

Compilation flags:

```
static
```

Template:

```
random_distinct_multiset(K,List,Multiset)
```

Mode and number of proofs:

```
random_distinct_multiset(+integer,+list,-list) - zero_or_one
```

---

sample_distinct_multisets/4

Returns SampleCount random distinct K-multisets with replacement of a list, sampled with replacement after deduplicating repeated values in the input list.

Compilation flags:

```
static
```

Template:

```
sample_distinct_multisets(K,List,SampleCount,Samples)
```

Mode and number of proofs:

```
sample_distinct_multisets(+integer,+list,+integer,-list) - zero_or_one
```

---

next_multiset/3

Returns the next distinct multiset value in lexicographic order induced by the first argument.

Compilation flags:

```
static
```

Template:

```
next_multiset(List,Multiset,Next)
```

Mode and number of proofs:

```
next_multiset(+list,+list,-list) - zero_or_one
```

---

previous_multiset/3

Returns the previous distinct multiset value in lexicographic order induced by the first argument.

Compilation flags:
`static`

Template:
`previous_multiset(List,Multiset,Previous)`
Mode and number of proofs:
`previous_multiset(+list,+list,-list)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.227 multivariate_distributions

object

1.227.1 multivariate_distributions(Random)

- `Random` - An object implementing the `sampling_protocol` protocol.

Multivariate probability distributions using an injected random source.

Availability:
`logtalk_load(multivariate_distributions(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *multivariate_distributions_protocol*
Uses:
integer
linear_algebra

list
numberlist
type

Remarks:
(none)

Inherited public predicates:
dirichlet/2 *dirichlet_density/3* *dirichlet_log_density/3* *dirichlet_samples/3* *logistic_normal/3* *logistic_normal/4* *logistic_normal_samples/4* *logistic_normal_samples/5* *mahalanobis_distance/4* *mahalanobis_distance/5* *multinomial/3* *multinomial_density/4* *multinomial_log_density/4* *multinomial_quantile/4* *multinomial_samples/4* *multivariate_normal/3* *multivariate_normal/4* *multivariate_normal_density/4* *multivariate_normal_density/5* *multivariate_normal_log_density/4* *multivariate_normal_log_density/5* *multivariate_normal_samples/4* *multivariate_normal_samples/5* *multivariate_t/4* *multivariate_t/5* *multivariate_t_density/5* *multivariate_t_density/6* *multivariate_t_log_density/5* *multivariate_t_log_density/6* *multivariate_t_samples/5* *multivariate_t_samples/6* *squared_mahalanobis_distance/4* *squared_mahalanobis_distance/5*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

multivariate_distributions_protocol, *linear_algebra*, *random(Algorithm)*, *fast_random(Algorithm)*, *backend_random*

protocol

1.227.2 multivariate_distributions_protocol

Multivariate probability distribution predicates.

Availability:
`logtalk_load(multivariate_distributions(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-12

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *multivariate_normal/3*
  - *multivariate_normal/4*
  - *multivariate_normal_samples/4*
  - *multivariate_normal_samples/5*
  - *multivariate_normal_density/4*
  - *multivariate_normal_density/5*
  - *multivariate_normal_log_density/4*
  - *multivariate_normal_log_density/5*
  - *squared_mahalanobis_distance/4*
  - *squared_mahalanobis_distance/5*
  - *mahalanobis_distance/4*
  - *mahalanobis_distance/5*
  - *multivariate_t/4*
  - *multivariate_t/5*
  - *multivariate_t_samples/5*
  - *multivariate_t_samples/6*
  - *multivariate_t_density/5*
  - *multivariate_t_density/6*
  - *multivariate_t_log_density/5*
  - *multivariate_t_log_density/6*
  - *logistic_normal/3*
  - *logistic_normal/4*
  - *logistic_normal_samples/4*

    - *logistic_normal_samples/5*
    - *dirichlet/2*
    - *dirichlet_samples/3*
    - *dirichlet_density/3*
    - *dirichlet_log_density/3*
    - *multinomial/3*
    - *multinomial_samples/4*
    - *multinomial_density/4*
    - *multinomial_log_density/4*
    - *multinomial_quantile/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

multivariate_normal/3

Returns a multivariate normally distributed random vector using the default numerical tolerance of 1.0e-12. Singular positive-semidefinite covariance matrices are supported.

Compilation flags:

`static`

Template:

`multivariate_normal(Mean,Covariance,Sample)`

Mode and number of proofs:

`multivariate_normal(+list(number),+list(list(number)),-list(float))` - `one_or_error`

Exceptions:

`Mean` is empty:

`domain_error(minimum_number_of_values(1),Mean)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Covariance` dimensions do not match the mean dimension:

`domain_error(covariance_dimensions(A),Covariance)`

`Covariance` is not symmetric:

`domain_error(symmetric_matrix,Covariance)`

`Covariance` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Covariance)`

multivariate_normal/4

Returns a multivariate normally distributed random vector using the given non-negative numerical tolerance. Singular positive-semidefinite covariance matrices are supported.

Compilation flags:

`static`

Template:

`multivariate_normal(Mean,Covariance,Tolerance,Sample)`

Mode and number of proofs:

`multivariate_normal(+list(number),+list(list(number)),+number,-list(float))` - `one_or_error`

Exceptions:

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Mean` is empty:

`domain_error(minimum_number_of_values(1),Mean)`

`Tolerance` is negative:

`domain_error(non_negative_number,Tolerance)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Covariance` dimensions do not match the mean dimension:

`domain_error(covariance_dimensions(A),Covariance)`

`Covariance` is not symmetric:

`domain_error(symmetric_matrix,Covariance)`

`Covariance` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Covariance)`

`Tolerance` is a variable:

`instantiation_error`

`Tolerance` is neither a variable nor a number:

`type_error(number,Tolerance)`

`Tolerance` is a nuber but not a non-negative number:

`domain_error(non_negative_number,Tolerance)`

multivariate_normal_samples/4

Returns the requested number of multivariate normally distributed random row vectors using the default numerical tolerance of 1.0e-12.

Compilation flags:

`static`

Template:

`multivariate_normal_samples(Count,Mean,Covariance,Samples)`

Mode and number of proofs:

`multivariate_normal_samples(+integer,+list(number),+list(list(number)),-list(list(float)))` - `one_or_error`

Exceptions:

`Count` is a variable:

`instantiation_error`

`Count` is neither a variable nor an integer:

`type_error(integer,Count)`

`Count` is an integer but not a non-negative integer:

`domain_error(non_negative_integer,Count)`

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Mean` is empty:

`domain_error(minimum_number_of_values(1),Mean)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Covariance` dimensions do not match the mean dimension:

`domain_error(covariance_dimensions(A),Covariance)`

`Covariance` is not symmetric:

`domain_error(symmetric_matrix,Covariance)`

`Covariance` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Covariance)`

multivariate_normal_samples/5

Returns the requested number of multivariate normally distributed random row vectors using the given non-negative numerical tolerance.

Compilation flags:

`static`

Template:

`multivariate_normal_samples(Count,Mean,Covariance,Tolerance,Samples)`

Mode and number of proofs:

`multivariate_normal_samples(+integer,+list(number),+list(list(number)),+number,-list(list(float)))` - `one_or_error`

Exceptions:

`Count` is a variable:

`instantiation_error`

`Count` is neither a variable nor an integer:

`type_error(integer,Count)`

`Count` is an integer but not a non-negative integer:

`domain_error(non_negative_integer,Count)`

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Tolerance` is a variable:

`instantiation_error`

`Tolerance` is neither a variable nor a number:

`type_error(number,Tolerance)`

`Tolerance` is a nuber but not a non-negative number:

`domain_error(non_negative_number,Tolerance)`

multivariate_normal_density/4

Computes the multivariate normal density at a point using the default numerical tolerance of 1.0e-12. For a singular covariance matrix, computes the density on its affine support and returns zero outside it.

Compilation flags:

`static`

Template:

`multivariate_normal_density(Point,Mean,Covariance,Density)`

Mode and number of proofs:

`multivariate_normal_density(+list(number),+list(number),+list(list(number)),-float)` - `one_or_error`

Exceptions:

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Mean` is empty:

`domain_error(minimum_number_of_values(1),Mean)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Covariance` dimensions do not match the mean dimension:

`domain_error(covariance_dimensions(A),Covariance)`

`Covariance` is not symmetric:

`domain_error(symmetric_matrix,Covariance)`

`Covariance` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Covariance)`

`Point` is a variable or a partial list:

`instantiation_error`

`Point` is neither a partial list nor a list:

`type_error(list,Point)`

An element `Element` of the `Point` list is neither a variable nor a number:

`type_error(number,Element)`

`Point` dimensions do not match the mean dimension:

`domain_error(point_dimensions(A),Point)`

multivariate_normal_density/5

Computes the multivariate normal density at a point using the given non-negative numerical tolerance. For a singular covariance matrix, computes the density on its affine support and returns zero outside it.

Compilation flags:

`static`

Template:

`multivariate_normal_density(Point,Mean,Covariance,Tolerance,Density)`

Mode and number of proofs:

`multivariate_normal_density(+list(number),+list(number),+list(list(number)),+number,-float)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:

`instantiation_error`

`Point` is neither a partial list nor a list:

`type_error(list,Point)`

An element `Element` of the `Point` list is neither a variable nor a number:

`type_error(number,Element)`

`Point` dimensions do not match the mean dimension:

`domain_error(point_dimensions(A),Point)`

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Mean` is empty:

`domain_error(minimum_number_of_values(1),Mean)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Covariance` dimensions do not match the mean dimension:

`domain_error(covariance_dimensions(A),Covariance)`

`Covariance` is not symmetric:

`domain_error(symmetric_matrix,Covariance)`

`Covariance` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Covariance)`

`Tolerance` is a variable:

`instantiation_error`

`Tolerance` is neither a variable nor a number:

`type_error(number,Tolerance)`

`Tolerance` is a nuber but not a non-negative number:

`domain_error(non_negative_number,Tolerance)`

multivariate_normal_log_density/4

Computes the multivariate normal log-density at a point using the default numerical tolerance of 1.0e-12. Returns the atom `negative_infinity` outside the affine support of a singular covariance matrix.

Compilation flags:

`static`

Template:

`multivariate_normal_log_density(Point,Mean,Covariance,LogDensity)`

Mode and number of proofs:

`multivariate_normal_log_density(+list(number),+list(number),+list(list(number)),-atomic)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:

`instantiation_error`

`Point` is neither a partial list nor a list:

`type_error(list,Point)`

An element `Element` of the `Point` list is neither a variable nor a number:

`type_error(number,Element)`

`Point` dimensions do not match the mean dimension:

`domain_error(point_dimensions(A),Point)`

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Mean` is empty:

`domain_error(minimum_number_of_values(1),Mean)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Covariance` dimensions do not match the mean dimension:

`domain_error(covariance_dimensions(A),Covariance)`

`Covariance` is not symmetric:

`domain_error(symmetric_matrix,Covariance)`

`Covariance` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Covariance)`

multivariate_normal_log_density/5

Computes the multivariate normal log-density at a point using the given non-negative numerical tolerance. Returns the atom `negative_infinity` outside the affine support of a singular covariance matrix.

Compilation flags:

`static`

Template:

`multivariate_normal_log_density(Point,Mean,Covariance,Tolerance,LogDensity)`

Mode and number of proofs:

`multivariate_normal_log_density(+list(number),+list(number),+list(list(number)),+number,-atomic)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:

`instantiation_error`

`Point` is neither a partial list nor a list:

`type_error(list,Point)`

An element `Element` of the `Point` list is neither a variable nor a number:

`type_error(number,Element)`

`Point` dimensions do not match the mean dimension:

`domain_error(point_dimensions(A),Point)`

`Tolerance` is negative:

`domain_error(non_negative_number,Tolerance)`

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Mean` is empty:

`domain_error(minimum_number_of_values(1),Mean)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Covariance` dimensions do not match the mean dimension:

`domain_error(covariance_dimensions(A),Covariance)`

`Covariance` is not symmetric:

`domain_error(symmetric_matrix,Covariance)`

`Covariance` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Covariance)`

`Tolerance` is a variable:

`instantiation_error`

`Tolerance` is neither a variable nor a number:

`type_error(number,Tolerance)`

`Tolerance` is a nuber but not a non-negative number:

`domain_error(non_negative_number,Tolerance)`

---

squared_mahalanobis_distance/4

Computes the squared Mahalanobis distance using the default numerical tolerance of 1.0e-12.

Compilation flags:

`static`

Template:

`squared_mahalanobis_distance(Point,Mean,Covariance,SquaredDistance)`

Mode and number of proofs:

`squared_mahalanobis_distance(+list(number),+list(number),+list(list(number)),-float)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:

`instantiation_error`

`Point` is neither a partial list nor a list:

`type_error(list,Point)`

An element `Element` of the `Point` list is neither a variable nor a number:

`type_error(number,Element)`

`Point` is outside the affine support of `Covariance`:

`domain_error(covariance_support,Point)`

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

---

squared_mahalanobis_distance/5

Computes the squared Mahalanobis distance using the given non-negative numerical tolerance.

Compilation flags:

`static`

Template:

`squared_mahalanobis_distance(Point,Mean,Covariance,Tolerance,SquaredDistance)`

Mode and number of proofs:

`squared_mahalanobis_distance(+list(number),+list(number),+list(list(number)),+number,-float)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:

`instantiation_error`

`Point` is neither a partial list nor a list:

`type_error(list,Point)`

An element `Element` of the `Point` list is neither a variable nor a number:

`type_error(number,Element)`

`Point` is outside the affine support of `Covariance`:

`domain_error(covariance_support,Point)`

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Tolerance` is a variable:

`instantiation_error`

`Tolerance` is neither a variable nor a number:

`type_error(number,Tolerance)`

`Tolerance` is a nuber but not a non-negative number:

`domain_error(non_negative_number,Tolerance)`

mahalanobis_distance/4

Computes the Mahalanobis distance using the default numerical tolerance of 1.0e-12.

Compilation flags:

```
static
```

Template:

```
mahalanobis_distance(Point,Mean,Covariance,Distance)
```

Mode and number of proofs:

```
mahalanobis_distance(+list(number),+list(number),+list(list(number)),-float) -
one_or_error
```

Exceptions:

`Point` is a variable or a partial list:

```
instantiation_error
```

`Point` is neither a partial list nor a list:

```
type_error(list,Point)
```

An element `Element` of the `Point` list is neither a variable nor a number:

```
type_error(number,Element)
```

`Point` is outside the affine support of `Covariance`:

```
domain_error(covariance_support,Point)
```

`Point` dimensions do not match the mean dimension:

```
domain_error(point_dimensions(A),Point)
```

`Mean` is a variable or a partial list:

```
instantiation_error
```

`Mean` is neither a partial list nor a list:

```
type_error(list,Mean)
```

An element `Element` of the `Mean` list is neither a variable nor a number:

```
type_error(number,Element)
```

`Covariance` is a variable or a partial list:

```
instantiation_error
```

`Covariance` is neither a partial list nor a list:

```
type_error(list(list(number)),Covariance)
```

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

```
type_error(list(number),Element)
```

---

mahalanobis_distance/5

Computes the Mahalanobis distance using the given non-negative numerical tolerance.

Compilation flags:

```
static
```

Template:

```
mahalanobis_distance(Point,Mean,Covariance,Tolerance,Distance)
```

Mode and number of proofs:

`mahalanobis_distance(+list(number),+list(number),+list(list(number)),+number,-float)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:
`instantiation_error`
`Point` is neither a partial list nor a list:
`type_error(list,Point)`
An element `Element` of the `Point` list is neither a variable nor a number:
`type_error(number,Element)`
`Point` is outside the affine support of `Covariance`:
`domain_error(covariance_support,Point)`
`Point` dimensions do not match the mean dimension:
`domain_error(point_dimensions(A),Point)`
`Mean` is a variable or a partial list:
`instantiation_error`
`Mean` is neither a partial list nor a list:
`type_error(list,Mean)`
An element `Element` of the `Mean` list is neither a variable nor a number:
`type_error(number,Element)`
`Covariance` is a variable or a partial list:
`instantiation_error`
`Covariance` is neither a partial list nor a list:
`type_error(list(list(number)),Covariance)`
An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:
`type_error(list(number),Element)`
`Tolerance` is a variable:
`instantiation_error`
`Tolerance` is neither a variable nor a number:
`type_error(number,Tolerance)`
`Tolerance` is a nuber but not a non-negative number:
`domain_error(non_negative_number,Tolerance)`

`multivariate_t/4`

Returns a multivariate Student's t distributed random vector using the default numerical tolerance of 1.0e-12.

Compilation flags:
`static`

Template:
`multivariate_t(DegreesOfFreedom,Location,Scale,Sample)`
Mode and number of proofs:
`multivariate_t(+positive_number,+list(number),+list(list(number)),-list(float))` - `one_or_error`

Exceptions:

`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor an integer:
`type_error(integer,DegreesOfFreedom)`
`DegreesOfFreedom` is not a positive integer:
`domain_error(positive_number,DegreesOfFreedom)`
`Location` is empty:
`domain_error(minimum_number_of_values(1),Location)`
`Scale` dimensions do not match the location dimension:
`domain_error(covariance_dimensions(A),Scale)`
`Scale` is not symmetric:
`domain_error(symmetric_matrix,Scale)`
`Scale` is not positive semidefinite:
`domain_error(positive_semidefinite_matrix,Scale)`

`multivariate_t/5`

Returns a multivariate Student's t distributed random vector using the given non-negative numerical tolerance.

Compilation flags:
`static`

Template:
`multivariate_t(DegreesOfFreedom,Location,Scale,Tolerance,Sample)`
Mode and number of proofs:
`multivariate_t(+positive_number,+list(number),+list(list(number)),+number,-list(float))` - `one_or_error`

Exceptions:
`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`
`Tolerance` is negative:
`domain_error(non_negative_number,Tolerance)`
`Location` is empty:
`domain_error(minimum_number_of_values(1),Location)`
`Scale` dimensions do not match the location dimension:
`domain_error(covariance_dimensions(A),Scale)`
`Scale` is not symmetric:
`domain_error(symmetric_matrix,Scale)`
`Scale` is not positive semidefinite:
`domain_error(positive_semidefinite_matrix,Scale)`
`Tolerance` is a variable:
`instantiation_error`

`Tolerance` is neither a variable nor a number:
`type_error(number,Tolerance)`
`Tolerance` is a nuber but not a non-negative number:
`domain_error(non_negative_number,Tolerance)`

`multivariate_t_samples/5`

Returns the requested number of multivariate Student's t distributed random row vectors using the default numerical tolerance of 1.0e-12.

Compilation flags:
`static`

Template:
`multivariate_t_samples(Count,DegreesOfFreedom,Location,Scale,Samples)`
Mode and number of proofs:
`multivariate_t_samples(+integer,+positive_number,+list(number),`
`+list(list(number)),-list(list(float)))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Count)`
`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`
`Location` is empty:
`domain_error(minimum_number_of_values(1),Location)`
`Scale` dimensions do not match the location dimension:
`domain_error(covariance_dimensions(A),Scale)`
`Scale` is not symmetric:
`domain_error(symmetric_matrix,Scale)`
`Scale` is not positive semidefinite:
`domain_error(positive_semidefinite_matrix,Scale)`

---

multivariate_t_samples/6

Returns the requested number of multivariate Student's t distributed random row vectors using the given non-negative numerical tolerance.

Compilation flags:

```
static
```

Template:

```
multivariate_t_samples(Count,DegreesOfFreedom,Location,Scale,Tolerance,Samples)
```

Mode and number of proofs:

```
multivariate_t_samples(+integer,+positive_number,+list(number),
+list(list(number)),+number,-list(list(float))) - one_or_error
```

Exceptions:

`Count` is a variable:

```
instantiation_error
```

`Count` is neither a variable nor an integer:

```
type_error(integer,Count)
```

`Count` is a negative integer:

```
domain_error(non_negative_integer,Count)
```

`DegreesOfFreedom` is a variable:

```
instantiation_error
```

`DegreesOfFreedom` is neither a variable nor a number:

```
type_error(number,DegreesOfFreedom)
```

`DegreesOfFreedom` is a number but not a positive number:

```
domain_error(positive_number,DegreesOfFreedom)
```

`Tolerance` is negative:

```
domain_error(non_negative_number,Tolerance)
```

`Location` is empty:

```
domain_error(minimum_number_of_values(1),Location)
```

`Scale` dimensions do not match the location dimension:

```
domain_error(covariance_dimensions(A),Scale)
```

`Scale` is not symmetric:

```
domain_error(symmetric_matrix,Scale)
```

`Scale` is not positive semidefinite:

```
domain_error(positive_semidefinite_matrix,Scale)
```

`Tolerance` is a variable:

```
instantiation_error
```

`Tolerance` is neither a variable nor a number:

```
type_error(number,Tolerance)
```

`Tolerance` is a nuber but not a non-negative number:

```
domain_error(non_negative_number,Tolerance)
```

multivariate_t_density/5

Computes the multivariate Student's t density using the default numerical tolerance of 1.0e-12. Returns zero outside the affine support of a singular scale matrix.

Compilation flags:

`static`

Template:

`multivariate_t_density(Point,DegreesOfFreedom,Location,Scale,Density)`

Mode and number of proofs:

`multivariate_t_density(+list(number),+positive_number,+list(number),+list(list(number)),-float)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:

`instantiation_error`

`Point` is neither a partial list nor a list:

`type_error(list,Point)`

An element `Element` of the `Point` list is neither a variable nor a number:

`type_error(number,Element)`

`Point` dimensions do not match the location dimension:

`domain_error(point_dimensions(A),Point)`

`DegreesOfFreedom` is a variable:

`instantiation_error`

`DegreesOfFreedom` is neither a variable nor a number:

`type_error(number,DegreesOfFreedom)`

`DegreesOfFreedom` is a number but not a positive number:

`domain_error(positive_number,DegreesOfFreedom)`

`Location` is empty:

`domain_error(minimum_number_of_values(1),Location)`

`Scale` dimensions do not match the location dimension:

`domain_error(covariance_dimensions(A),Scale)`

`Scale` is not symmetric:

`domain_error(symmetric_matrix,Scale)`

`Scale` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Scale)`

---

multivariate_t_density/6

Computes the multivariate Student's t density using the given non-negative numerical tolerance. Returns zero outside the affine support of a singular scale matrix.

Compilation flags:

`static`

Template:

`multivariate_t_density(Point,DegreesOfFreedom,Location,Scale,Tolerance,Density)`

Mode and number of proofs:

`multivariate_t_density(+list(number),+positive_number,+list(number),+list(list(number)),+number,-float)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:

`instantiation_error`

`Point` is neither a partial list nor a list:

`type_error(list,Point)`

An element `Element` of the `Point` list is neither a variable nor a number:

`type_error(number,Element)`

`Point` dimensions do not match the location dimension:

`domain_error(point_dimensions(A),Point)`

`DegreesOfFreedom` is a variable:

`instantiation_error`

`DegreesOfFreedom` is neither a variable nor a number:

`type_error(number,DegreesOfFreedom)`

`DegreesOfFreedom` is a number but not a positive number:

`domain_error(positive_number,DegreesOfFreedom)`

`Location` is empty:

`domain_error(minimum_number_of_values(1),Location)`

`Scale` dimensions do not match the location dimension:

`domain_error(covariance_dimensions(A),Scale)`

`Scale` is not symmetric:

`domain_error(symmetric_matrix,Scale)`

`Scale` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Scale)`

`Tolerance` is a variable:

`instantiation_error`

`Tolerance` is neither a variable nor a number:

`type_error(number,Tolerance)`

`Tolerance` is a nuber but not a non-negative number:

`domain_error(non_negative_number,Tolerance)`

`multivariate_t_log_density/5`

Computes the multivariate Student's t log-density using the default numerical tolerance of 1.0e-12. Returns `negative_infinity` outside singular affine support.

Compilation flags:

`static`

Template:

`multivariate_t_log_density(Point,DegreesOfFreedom,Location,Scale,LogDensity)`

Mode and number of proofs:

`multivariate_t_log_density(+list(number),+positive_number,+list(number),+list(list(number)),-atomic)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:

```
instantiation_error
```

`Point` is neither a partial list nor a list:

```
type_error(list,Point)
```

An element `Element` of the `Point` list is neither a variable nor a number:

```
type_error(number,Element)
```

`Point` dimensions do not match the location dimension:

```
domain_error(point_dimensions(A),Point)
```

`DegreesOfFreedom` is a variable:

```
instantiation_error
```

`DegreesOfFreedom` is neither a variable nor a number:

```
type_error(number,DegreesOfFreedom)
```

`DegreesOfFreedom` is a number but not a positive number:

```
domain_error(positive_number,DegreesOfFreedom)
```

`Location` is empty:

```
domain_error(minimum_number_of_values(1),Location)
```

`Scale` dimensions do not match the location dimension:

```
domain_error(covariance_dimensions(A),Scale)
```

`Scale` is not symmetric:

```
domain_error(symmetric_matrix,Scale)
```

`Scale` is not positive semidefinite:

```
domain_error(positive_semidefinite_matrix,Scale)
```

`multivariate_t_log_density/6`

Computes the multivariate Student's t log-density using the given non-negative numerical tolerance. Returns `negative_infinity` outside singular affine support.

Compilation flags:

```
static
```

Template:

```
multivariate_t_log_density(Point,DegreesOfFreedom,Location,Scale,Tolerance,
LogDensity)
```

Mode and number of proofs:

```
multivariate_t_log_density(+list(number),+positive_number,+list(number),
+list(list(number)),+number,-atomic) - one_or_error
```

Exceptions:

`Point` is a variable or a partial list:

```
instantiation_error
```

`Point` is neither a partial list nor a list:

```
type_error(list,Point)
```

An element `Element` of the `Point` list is neither a variable nor a number:

```
type_error(number,Element)
```

`Point` dimensions do not match the location dimension:

`domain_error(point_dimensions(A),Point)`

`DegreesOfFreedom` is a variable:

`instantiation_error`

`DegreesOfFreedom` is neither a variable nor a number:

`type_error(number,DegreesOfFreedom)`

`DegreesOfFreedom` is a number but not a positive number:

`domain_error(positive_number,DegreesOfFreedom)`

`Location` is empty:

`domain_error(minimum_number_of_values(1),Location)`

`Scale` dimensions do not match the location dimension:

`domain_error(covariance_dimensions(A),Scale)`

`Scale` is not symmetric:

`domain_error(symmetric_matrix,Scale)`

`Scale` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Scale)`

`Tolerance` is a variable:

`instantiation_error`

`Tolerance` is neither a variable nor a number:

`type_error(number,Tolerance)`

`Tolerance` is a nuber but not a non-negative number:

`domain_error(non_negative_number,Tolerance)`

`logistic_normal/3`

Returns an additive-log-ratio logistic-normal random vector using the default numerical tolerance of 1.0e-12. A latent vector of length d maps to a simplex vector of length d+1 using the final component as reference.

Compilation flags:

`static`

Template:

`logistic_normal(Mean,Covariance,Sample)`

Mode and number of proofs:

`logistic_normal(+list(number),+list(list(number)),-list(float))` - `one_or_error`

Exceptions:

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Mean` is empty:

`domain_error(minimum_number_of_values(1),Mean)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`
An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:
`type_error(list(number),Element)`
`Covariance` dimensions do not match the mean dimension:
`domain_error(covariance_dimensions(A),Covariance)`
`Covariance` is not symmetric:
`domain_error(symmetric_matrix,Covariance)`
`Covariance` is not positive semidefinite:
`domain_error(positive_semidefinite_matrix,Covariance)`

`logistic_normal/4`

Returns an additive-log-ratio logistic-normal random vector using the given non-negative numerical tolerance.

Compilation flags:
`static`

Template:
`logistic_normal(Mean,Covariance,Tolerance,Sample)`
Mode and number of proofs:
`logistic_normal(+list(number),+list(list(number)),+number,-list(float))` - `one_or_error`

Exceptions:
`Mean` is a variable or a partial list:
`instantiation_error`
`Mean` is neither a partial list nor a list:
`type_error(list,Mean)`
An element `Element` of the `Mean` list is neither a variable nor a number:
`type_error(number,Element)`
`Mean` is empty:
`domain_error(minimum_number_of_values(1),Mean)`
`Covariance` is a variable or a partial list:
`instantiation_error`
`Covariance` is neither a partial list nor a list:
`type_error(list(list(number)),Covariance)`
An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:
`type_error(list(number),Element)`
`Covariance` dimensions do not match the mean dimension:
`domain_error(covariance_dimensions(A),Covariance)`
`Covariance` is not symmetric:
`domain_error(symmetric_matrix,Covariance)`
`Covariance` is not positive semidefinite:
`domain_error(positive_semidefinite_matrix,Covariance)`
`Tolerance` is a variable:
`instantiation_error`
`Tolerance` is neither a variable nor a number:

`type_error(number,Tolerance)`

`Tolerance` is a nuber but not a non-negative number:

`domain_error(non_negative_number,Tolerance)`

`logistic_normal_samples/4`

Returns the requested number of additive-log-ratio logistic-normal random row vectors using the default numerical tolerance of 1.0e-12.

Compilation flags:

`static`

Template:

`logistic_normal_samples(Count,Mean,Covariance,Samples)`

Mode and number of proofs:

`logistic_normal_samples(+integer,+list(number),+list(list(number)),-list(list(float)))` - `one_or_error`

Exceptions:

`Count` is a variable:

`instantiation_error`

`Count` is neither a variable nor an integer:

`type_error(integer,Count)`

`Count` is an integer but not a non-negative integer:

`domain_error(non_negative_integer,Count)`

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Mean` is empty:

`domain_error(minimum_number_of_values(1),Mean)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Covariance` dimensions do not match the mean dimension:

`domain_error(covariance_dimensions(A),Covariance)`

`Covariance` is not symmetric:

`domain_error(symmetric_matrix,Covariance)`

`Covariance` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Covariance)`

---

logistic_normal_samples/5

Returns the requested number of additive-log-ratio logistic-normal random row vectors using the given non-negative numerical tolerance.

Compilation flags:

`static`

Template:

`logistic_normal_samples(Count,Mean,Covariance,Tolerance,Samples)`

Mode and number of proofs:

`logistic_normal_samples(+integer,+list(number),+list(list(number)),+number,-list(list(float)))` - `one_or_error`

Exceptions:

`Count` is a variable:

`instantiation_error`

`Count` is neither a variable nor an integer:

`type_error(integer,Count)`

`Count` is an integer but not a non-negative integer:

`domain_error(non_negative_integer,Count)`

`Mean` is a variable or a partial list:

`instantiation_error`

`Mean` is neither a partial list nor a list:

`type_error(list,Mean)`

An element `Element` of the `Mean` list is neither a variable nor a number:

`type_error(number,Element)`

`Mean` is empty:

`domain_error(minimum_number_of_values(1),Mean)`

`Covariance` is a variable or a partial list:

`instantiation_error`

`Covariance` is neither a partial list nor a list:

`type_error(list(list(number)),Covariance)`

An element `Element` of the `Covariance` list is neither a variable nor a list of numbers:

`type_error(list(number),Element)`

`Covariance` dimensions do not match the mean dimension:

`domain_error(covariance_dimensions(A),Covariance)`

`Covariance` is not symmetric:

`domain_error(symmetric_matrix,Covariance)`

`Covariance` is not positive semidefinite:

`domain_error(positive_semidefinite_matrix,Covariance)`

`Tolerance` is a variable:

`instantiation_error`

`Tolerance` is neither a variable nor a number:

`type_error(number,Tolerance)`

`Tolerance` is a nuber but not a non-negative number:

`domain_error(non_negative_number,Tolerance)`

dirichlet/2

Returns a Dirichlet distributed random vector on the simplex.

Compilation flags:
`static`

Template:
`dirichlet(Alphas,Sample)`
Mode and number of proofs:
`dirichlet(+list(positive_number),-list(float))` - `one_or_error`

Exceptions:
`Alphas` is a variable or a partial list:
`instantiation_error`
`Alphas` is neither a partial list nor a list:
`type_error(list,Alphas)`
An element `Element` of the `Alphas` list is neither a variable nor a number:
`type_error(number,Element)`
An element `Element` of the `Alphas` list is not positive:
`domain_error(positive_number,Element)`
`Alphas` has fewer than two elements:
`domain_error(minimum_number_of_values(2),Alphas)`

---

dirichlet_samples/3

Returns the requested number of Dirichlet distributed random vectors.

Compilation flags:
`static`

Template:
`dirichlet_samples(Count,Alphas,Samples)`
Mode and number of proofs:
`dirichlet_samples(+integer,+list(positive_number),-list(list(float)))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Count)`
`Alphas` is a variable or a partial list:
`instantiation_error`
`Alphas` is neither a partial list nor a list:

`type_error(list,Alphas)`

An element `Element` of the `Alphas` list is neither a variable nor a number:

`type_error(number,Element)`

An element `Element` of the `Alphas` list is not positive:

`domain_error(positive_number,Element)`

`Alphas` has fewer than two elements:

`domain_error(minimum_number_of_values(2),Alphas)`

---

dirichlet_density/3

Computes the Dirichlet density at a point on the simplex. At a boundary, returns the atom `positive_infinity` when all non-unit alphas corresponding to zero components are smaller than one, zero when they are all greater than one, or `undefined` when both cases occur.

Compilation flags:

`static`

Template:

`dirichlet_density(Point,Alphas,Density)`

Mode and number of proofs:

`dirichlet_density(+list(number),+list(positive_number),-atomic)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:

`instantiation_error`

`Point` is neither a partial list nor a list:

`type_error(list,Point)`

An element `Element` of the `Point` list is neither a variable nor a number:

`type_error(number,Element)`

`Alphas` is a variable or a partial list:

`instantiation_error`

`Alphas` is neither a partial list nor a list:

`type_error(list,Alphas)`

An element `Element` of the `Alphas` list is neither a variable nor a number:

`type_error(number,Element)`

An element `Element` of the `Alphas` list is not positive:

`domain_error(positive_number,Element)`

`Point` and `Alphas` have different lengths:

`domain_error(dimension_mismatch,Point)`

`Alphas` has fewer than two elements:

`domain_error(minimum_number_of_values(2),Alphas)`

---

dirichlet_log_density/3

Computes the Dirichlet log-density at a point on the simplex. At a boundary, returns the atom `positive_infinity` when all non-unit alphas corresponding to zero components are smaller than one, `negative_infinity` when they are all greater than one, or `undefined` when both cases occur.

Compilation flags:

`static`

Template:

`dirichlet_log_density(Point,Alphas,LogDensity)`

Mode and number of proofs:

`dirichlet_log_density(+list(number),+list(positive_number),-atomic)` - `one_or_error`

Exceptions:

`Point` is a variable or a partial list:

`instantiation_error`

`Point` is neither a partial list nor a list:

`type_error(list,Point)`

An element `Element` of the `Point` list is neither a variable nor a number:

`type_error(number,Element)`

`Alphas` is a variable or a partial list:

`instantiation_error`

`Alphas` is neither a partial list nor a list:

`type_error(list,Alphas)`

An element `Element` of the `Alphas` list is neither a variable nor a number:

`type_error(number,Element)`

An element `Element` of the `Alphas` list is not positive:

`domain_error(positive_number,Element)`

`Point` and `Alphas` have different lengths:

`domain_error(dimension_mismatch,Point)`

`Alphas` has fewer than two elements:

`domain_error(minimum_number_of_values(2),Alphas)`

---

multinomial/3

Returns a multinomial distributed random count vector.

Compilation flags:

`static`

Template:

`multinomial(Trials,Probabilities,Counts)`

Mode and number of proofs:

`multinomial(+non_negative_integer,+list(probability),-list(non_negative_integer))` - `one_or_error`

Exceptions:

`Trials` is a variable:
`instantiation_error`
`Trials` is neither a variable nor an integer:
`type_error(integer,Trials)`
`Trials` is a negative integer:
`domain_error(non_negative_integer,Trials)`
`Probabilities` is a variable or a partial list:
`instantiation_error`
`Probabilities` is neither a partial list nor a list:
`type_error(list,Probabilities)`
An element `Element` of the `Probabilities` list is neither a variable nor a float:
`type_error(float,Element)`
An element `Element` of the `Probabilities` list is a float but not a probability:
`domain_error(probability,Element)`
`Probabilities` is empty:
`domain_error(minimum_number_of_values(1),Probabilities)`
`Probabilities` do not sum to one:
`domain_error(probability_distribution,Probabilities)`

`multinomial_samples/4`

Returns the requested number of multinomial distributed random count vectors.

Compilation flags:
`static`

Template:
`multinomial_samples(Count,Trials,Probabilities,Samples)`
Mode and number of proofs:
`multinomial_samples(+integer,+non_negative_integer,+list(probability),-list(list(non_negative_integer)))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Count)`
`Trials` is a variable:
`instantiation_error`
`Trials` is neither a variable nor an integer:
`type_error(integer,Trials)`
`Trials` is a negative integer:
`domain_error(non_negative_integer,Trials)`
`Probabilities` is a variable or a partial list:
`instantiation_error`

`Probabilities` is neither a partial list nor a list:
`type_error(list,Probabilities)`
An element `Element` of the `Probabilities` list is neither a variable nor a float:
`type_error(float,Element)`
An element `Element` of the `Probabilities` list is a float but not a probability:
`domain_error(probability,Element)`
`Probabilities` is empty:
`domain_error(minimum_number_of_values(1),Probabilities)`
`Probabilities` do not sum to one:
`domain_error(probability_distribution,Probabilities)`

`multinomial_density/4`

Computes the multinomial probability mass at the given count vector.

Compilation flags:
`static`

Template:
`multinomial_density(Counts,Trials,Probabilities,Density)`
Mode and number of proofs:
`multinomial_density(+list(non_negative_integer),+non_negative_integer,`
`+list(probability),-float)` - `one_or_error`

Exceptions:
`Counts` is a variable or a partial list:
`instantiation_error`
`Counts` is neither a partial list nor a list:
`type_error(list,Counts)`
An element `Element` of the `Counts` list is neither a variable nor an integer:
`type_error(integer,Element)`
An element `Element` of the `Counts` list is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Element)`
`Trials` is a variable:
`instantiation_error`
`Trials` is neither a variable nor an integer:
`type_error(integer,Trials)`
`Trials` is a negative integer:
`domain_error(non_negative_integer,Trials)`
`Probabilities` is a variable or a partial list:
`instantiation_error`
`Probabilities` is neither a partial list nor a list:
`type_error(list,Probabilities)`
An element `Element` of the `Probabilities` list is neither a variable nor a float:
`type_error(float,Element)`
An element `Element` of the `Probabilities` list is a float but not a probability:
`domain_error(probability,Element)`

`Probabilities` is empty:
`domain_error(minimum_number_of_values(1),Probabilities)`
`Probabilities` do not sum to one:
`domain_error(probability_distribution,Probabilities)`
`Counts` and `Probabilities` have different lengths:
`domain_error(dimension_mismatch,Counts)`

---

multinomial_log_density/4

Computes the multinomial log-probability mass at the given count vector. Returns the atom `negative_infinity` for an impossible count vector.

Compilation flags:
`static`

Template:
`multinomial_log_density(Counts,Trials,Probabilities,LogDensity)`
Mode and number of proofs:
`multinomial_log_density(+list(non_negative_integer),+non_negative_integer,+list(probability),-atomic)` - `one_or_error`

Exceptions:
`Counts` is a variable or a partial list:
`instantiation_error`
`Counts` is neither a partial list nor a list:
`type_error(list,Counts)`
An element `Element` of the `Counts` list is neither a variable nor an integer:
`type_error(integer,Element)`
An element `Element` of the `Counts` list is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Element)`
`Trials` is a variable:
`instantiation_error`
`Trials` is neither a variable nor an integer:
`type_error(integer,Trials)`
`Trials` is a negative integer:
`domain_error(non_negative_integer,Trials)`
`Probabilities` is a variable or a partial list:
`instantiation_error`
`Probabilities` is neither a partial list nor a list:
`type_error(list,Probabilities)`
An element `Element` of the `Probabilities` list is neither a variable nor a float:
`type_error(float,Element)`
An element `Element` of the `Probabilities` list is a float but not a probability:
`domain_error(probability,Element)`
`Probabilities` is empty:
`domain_error(minimum_number_of_values(1),Probabilities)`
`Probabilities` do not sum to one:

`domain_error(probability_distribution,Probabilities)`

`Counts` and `Probabilities` have different lengths:

`domain_error(dimension_mismatch,Counts)`

`multinomial_quantile/4`

Computes a multinomial quantile (count vector) for a probability strictly between zero and one. Count vectors are ordered by decreasing probability mass and then by increasing lexicographic order for log-probabilities equal within a relative tolerance of 1.0e-12. The returned vector is the first whose cumulative probability mass is at least the requested probability. The exact computation is limited to 100000 count vectors.

Compilation flags:

`static`

Template:

`multinomial_quantile(Probability,Trials,Probabilities,Quantile)`

Mode and number of proofs:

`multinomial_quantile(+open_probability,+non_negative_integer,+list(probability),-list(non_negative_integer))` - `one_or_error`

Exceptions:

`Probability` is a variable:

`instantiation_error`

`Probability` is neither a variable nor a float:

`type_error(float,Probability)`

`Probability` is a float but not strictly between zero and one:

`domain_error(open_probability,Probability)`

`Trials` is a variable:

`instantiation_error`

`Trials` is neither a variable nor an integer:

`type_error(integer,Trials)`

`Trials` is a negative integer:

`domain_error(non_negative_integer,Trials)`

`Probabilities` is a variable or a partial list:

`instantiation_error`

`Probabilities` is neither a partial list nor a list:

`type_error(list,Probabilities)`

An element `Element` of the `Probabilities` list is neither a variable nor a float:

`type_error(float,Element)`

An element `Element` of the `Probabilities` list is a float but not a probability:

`domain_error(probability,Element)`

`Probabilities` is empty:

`domain_error(minimum_number_of_values(1),Probabilities)`

`Probabilities` do not sum to one:

`domain_error(probability_distribution,Probabilities)`

The number of count vectors exceeds the supported limit:

`resource_error(multinomial_quantile_compositions)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

multivariate_distributions(Random), *linear_algebra*, *sampling_protocol*

1.228 mutation_testing

object

1.228.1 arithmetic_operator_replacement(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting clauses to mutate.
- `ClauseIndex` - 1-based clause index for the selected mutation (equal to `Occurrence` for this mutator).
- `Occurrence` - 1-based mutation occurrence index to target within selected predicate clauses.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `arithmetic_operator_replacement` mutator for matching predicate clauses.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
`public` *clause_mutator_protocol*
Imports:
`public` *mutator_common*

Remarks:
(none)

Inherited public predicates:

coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.228.2 body_goal_negation(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting clauses to mutate.
- `ClauseIndex` - 1-based clause index for the selected mutation (equal to `Occurrence` for this mutator).
- `Occurrence` - 1-based mutation occurrence index to target within selected predicate clauses.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the body_goal_negation mutator by negating matching predicate clause bodies.

Availability:

```
logtalk_load(mutation_testing(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public expanding
public clause_mutator_protocol
```

Imports:

`public` *mutator_common*

Remarks:
(none)

Inherited public predicates:
coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.228.3 `clause_mutator_protocol`

Marker protocol for clause and grammar-rule mutators.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:
`static`

Extends:
`public` *mutator_protocol*

Remarks:
(none)

Inherited public predicates:

coverage_clause_mutator/0 *mutation/2* *reset/0*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.228.4 clauses_reordering(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting clauses to mutate.
- `ClauseIndex` - 1-based clause index selecting which clause is swapped with its successor (last swaps with first).
- `Occurrence` - 1-based clause index selecting which clause is swapped with its successor (last swaps with first).
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `clauses_reordering` mutator by reordering the clauses of a non-discontiguous predicate or non-terminal definition.

Availability:

`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-06-07

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *expanding*
`public` *clause_mutator_protocol*

Imports:

`public` *mutator_common*

Uses:

list

Remarks:

(none)

Inherited public predicates:

coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *clauses_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

clauses_/1

Table of parsed clauses.

Compilation flags:

`dynamic`

Template:

`clauses_(Clause)`

Mode and number of proofs:

`clauses_(?clause)` - `zero_or_more`

Operators

(none)
protocol

1.228.5 directive_mutator_protocol

Marker protocol for directive mutators.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:
`static`

Extends:
`public` *mutator_protocol*

Remarks:
(none)

Inherited public predicates:
coverage_clause_mutator/0 *mutation/2* *reset/0*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.228.6 `fail_insertion(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)`

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting clauses to mutate.
- `ClauseIndex` - 1-based clause index for the selected mutation (equal to `Occurrence` for this mutator).
- `Occurrence` - 1-based mutation occurrence index to target within selected predicate clauses.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `fail_insertion` mutator by inserting fail at deterministic body positions for matching predicate clauses.

Availability:

```
logtalk_load(mutation_testing(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public expanding
public clause_mutator_protocol
```

Imports:

```
public mutator_common
```

Remarks:

(none)

Inherited public predicates:

coverage_clause_mutator/0 goal_expansion/2 mutation/2 reset/0 term_expansion/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.228.7 head_arguments_mutation(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting clauses to mutate.
- `ClauseIndex` - 1-based clause index for the selected mutation (equal to `Occurrence` for this mutator).
- `Occurrence` - 1-based mutation occurrence index selecting one compile-time bound head argument to mutate.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `head_arguments_mutation` mutator by mutating one compile-time bound predicate/non-terminal head argument using the `type::mutation/3` predicate.

Availability:

`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *expanding*
`public` *clause_mutator_protocol*

Imports:

`public` *mutator_common*

Uses:

type

Remarks:

(none)

Inherited public predicates:

coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.228.8 `head_arguments_reordering(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)`

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting clauses to mutate.
- `ClauseIndex` - 1-based clause index for the selected mutation (equal to `Occurrence` for this mutator).
- `Occurrence` - 1-based mutation occurrence index to target within selected predicate clauses.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `head_arguments_reordering` mutator by swapping the first two arguments in matching rule or grammar rule heads.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
`public` *clause_mutator_protocol*
Imports:
`public` *mutator_common*
Uses:

list

Remarks:
(none)

Inherited public predicates:
coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.228.9 `mutation_testing`

Mutation testing tool.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:4:1
Date: 2026-11-18

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Provides:
logtalk::message_hook/4
Uses:

fast_random(Algorithm)
json(ObjectRepresentation,PairRepresentation,StringRepresentation)
list
logtalk
os
term_io
type
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - `library/1`
  - `library/2`
  - `directory/1`
  - `directory/2`
  - `entity/1`
  - `entity/2`
  - `predicate/2`
  - `predicate/3`
  - `report_entity/3`
  - `report_predicate/4`
  - `report_library/3`
  - `report_directory/3`
  - `format_report/3`
  - `format_report/2`
  - `format_report/1`
  - `entity_mutants/2`
  - `entity_mutants/3`
  - `predicate_mutants/3`
  - `predicate_mutants/4`
  - `library_mutants/2`
  - `library_mutants/3`
  - `directory_mutants/2`
  - `directory_mutants/3`
- *Protected predicates*
- *Private predicates*
  - `probe_mutation_happened_/0`
  - `probing_/0`
  - `capturing_mutated_terms_/0`
  - `captured_mutated_terms_/5`
  - `baseline_coverage_/4`
- *Operators*

Public predicates

library/1

Runs mutation testing for all loaded entities from a given library using default options.

Compilation flags:
`static`

Template:
`library(Library)`
Mode and number of proofs:
`library(+atom)` - `zero_or_one`

---

library/2

Runs mutation testing for all loaded entities from a given library using the given options.

Compilation flags:
`static`

Template:
`library(Library,Options)`
Mode and number of proofs:
`library(+atom,+list(compound))` - `zero_or_one`

---

directory/1

Runs mutation testing for all loaded entities from a given directory using default options.

Compilation flags:
`static`

Template:
`directory(Directory)`
Mode and number of proofs:
`directory(+atom)` - `zero_or_one`

---

directory/2

Runs mutation testing for all loaded entities from a given directory using the given options.

Compilation flags:
`static`

Template:
`directory(Directory,Options)`
Mode and number of proofs:
`directory(+atom,+list(compound))` - `zero_or_one`

---

entity/1

Runs mutation testing for a loaded entity using default options.

Compilation flags:
`static`

Template:
`entity(Entity)`
Mode and number of proofs:
`entity(+entity_identifier)` - `zero_or_one`

---

entity/2

Runs mutation testing for a loaded entity using the given options.

Compilation flags:
`static`

Template:
`entity(Entity,Options)`
Mode and number of proofs:
`entity(+entity_identifier,+list(compound))` - `zero_or_one`

---

predicate/2

Runs mutation testing for a loaded entity predicate using default options.

Compilation flags:
```
static
```

Template:
```
predicate(Entity,Predicate)
```
Mode and number of proofs:
```
predicate(+entity_identifier,+predicate_indicator) - zero_or_one
```

---

predicate/3

Runs mutation testing for a loaded entity predicate using the given options.

Compilation flags:
```
static
```

Template:
```
predicate(Entity,Predicate,Options)
```
Mode and number of proofs:
```
predicate(+entity_identifier,+predicate_indicator,+list(compound)) - zero_or_one
```

---

report_entity/3

Runs mutation testing for an entity and returns a structured report term.

Compilation flags:
```
static
```

Template:
```
report_entity(Entity,Report,Options)
```
Mode and number of proofs:
```
report_entity(+entity_identifier,-compound,+list(compound)) - zero_or_one
```

---

report_predicate/4

Runs mutation testing for an entity predicate and returns a structured report term.

Compilation flags:
`static`

Template:
`report_predicate(Entity,Predicate,Report,Options)`
Mode and number of proofs:
`report_predicate(+entity_identifier,+predicate_indicator,-compound,+list(compound))` - `zero_or_one`

---

report_library/3

Runs mutation testing for loaded entities from a library and returns structured report terms.

Compilation flags:
`static`

Template:
`report_library(Library,Report,Options)`
Mode and number of proofs:
`report_library(+atom,-compound,+list(compound))` - `zero_or_one`

---

report_directory/3

Runs mutation testing for loaded entities from a directory and returns structured report terms.

Compilation flags:
`static`

Template:
`report_directory(Directory,Report,Options)`
Mode and number of proofs:
`report_directory(+atom,-compound,+list(compound))` - `zero_or_one`

---

format_report/3

Formats a mutation testing report term to the given stream using the given format.

Compilation flags:

```
static
```

Template:

```
format_report(Stream,Format,Report)
```

Mode and number of proofs:

```
format_report(+stream_or_alias,+atom,+compound) - one
```

---

format_report/2

Formats a mutation testing report term to the current output stream using the given format.

Compilation flags:

```
static
```

Template:

```
format_report(Format,Report)
```

Mode and number of proofs:

```
format_report(+atom,+compound) - one
```

---

format_report/1

Formats a mutation testing report term to the current output stream using the text format.

Compilation flags:

```
static
```

Template:

```
format_report(Report)
```

Mode and number of proofs:

```
format_report(+compound) - one
```

---

entity_mutants/2

Returns the deterministic list of mutants for an entity using default options.

Compilation flags:
`static`

Template:
`entity_mutants(Entity,Mutants)`
Mode and number of proofs:
`entity_mutants(+entity_identifier,-list(compound))` - `zero_or_one`

---

entity_mutants/3

Returns the deterministic list of mutants for an entity using the given options.

Compilation flags:
`static`

Template:
`entity_mutants(Entity,Mutants,Options)`
Mode and number of proofs:
`entity_mutants(+entity_identifier,-list(compound),+list(compound))` - `zero_or_one`

---

predicate_mutants/3

Returns the deterministic list of mutants for an entity predicate using default options.

Compilation flags:
`static`

Template:
`predicate_mutants(Entity,Predicate,Mutants)`
Mode and number of proofs:
`predicate_mutants(+entity_identifier,+predicate_indicator,-list(compound))` - `zero_or_one`

---

predicate_mutants/4

Returns the deterministic list of mutants for an entity predicate using the given options.

Compilation flags:
```
static
```

Template:
```
predicate_mutants(Entity,Predicate,Mutants,Options)
```
Mode and number of proofs:
```
predicate_mutants(+entity_identifier,+predicate_indicator,-list(compound),
+list(compound)) - zero_or_one
```

---

library_mutants/2

Returns the deterministic list of mutants for loaded entities from a given library using default options.

Compilation flags:
```
static
```

Template:
```
library_mutants(Library,Mutants)
```
Mode and number of proofs:
```
library_mutants(+atom,-list(compound)) - zero_or_one
```

---

library_mutants/3

Returns the deterministic list of mutants for loaded entities from a given library using the given options.

Compilation flags:
```
static
```

Template:
```
library_mutants(Library,Mutants,Options)
```
Mode and number of proofs:
```
library_mutants(+atom,-list(compound),+list(compound)) - zero_or_one
```

---

directory_mutants/2

Returns the deterministic list of mutants for loaded entities from a given directory using default options.

Compilation flags:
```
static
```

Template:
```
directory_mutants(Directory,Mutants)
```
Mode and number of proofs:
```
directory_mutants(+atom,-list(compound)) - zero_or_one
```

---

directory_mutants/3

Returns the deterministic list of mutants for loaded entities from a given directory using the given options.

Compilation flags:
```
static
```

Template:
```
directory_mutants(Directory,Mutants,Options)
```
Mode and number of proofs:
```
directory_mutants(+atom,-list(compound),+list(compound)) - zero_or_one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

probe_mutation_happened_/0

True iff a mutation happened.

Compilation flags:
```
dynamic
```

Mode and number of proofs:
```
probe_mutation_happened_ - zero_or_one
```

---

probing_/0

True iff we are currently probing for mutations (suppresses printing).

Compilation flags:
dynamic

Mode and number of proofs:
probing_ - zero_or_one

---

capturing_mutated_terms_/0

True iff we are capturing original and mutated terms while formatting reports.

Compilation flags:
dynamic

Mode and number of proofs:
capturing_mutated_terms_ - zero_or_one

---

captured_mutated_terms_/5

Captured original and mutated terms, variable names, and source location for one mutant.

Compilation flags:
dynamic

Template:
captured_mutated_terms_(Original,Mutation,Variables,File,Lines)
Mode and number of proofs:
captured_mutated_terms_(-callable,-callable,-list,-atom,-compound) - zero_or_one

---

baseline_coverage_/4

Coverage baseline cache for an entity and predicate: covered clauses and total clauses.

Compilation flags:
dynamic

Template:
baseline_coverage_(Entity,Predicate,CoveredClauses,TotalClauses)
Mode and number of proofs:

`baseline_coverage_(?entity_identifier,?predicate_indicator,?list(integer),?integer)` - `zero_or_more`

Operators

(none)

category

1.228.10 `mutation_testing_messages`

Default message translations for the `mutation_testing` tool.

Availability:

`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-06

Compilation flags:

`static`

Provides:

logtalk::message_prefix_stream/4
logtalk::message_tokens//2

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.228.11 `mutator_common`

Mutator common predicate utilities.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:
`static`

Implements:
`public` *mutator_protocol*
Uses:
logtalk

Remarks:
(none)

Inherited public predicates:
coverage_clause_mutator/0 *mutation/2* *reset/0*

- *Public predicates*
- *Protected predicates*
  - *print_mutation/3*
  - *target_predicate/3*
  - *target_predicate_clause_index/4*
  - *target_scope_directive/3*
  - *target_scope_directive_index/4*
  - *target_predicate_directive/3*
  - *target_predicate_directive_index/4*
  - *target_uses_directive/3*
  - *target_uses_directive_index/4*

    - *next_occurrence/1*
- *Private predicates*
    - *current_predicate_clause_index_/2*
    - *update_target_predicate_clause_index_/2*
    - *current_scope_directive_index_/2*
    - *update_target_scope_directive_index_/2*
    - *current_predicate_directive_index_/2*
    - *update_target_predicate_directive_index_/2*
    - *current_uses_directive_index_/2*
    - *update_target_uses_directive_index_/2*
    - *seen_/1*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

print_mutation/3

Prints a term and its mutation when `Flag` is true. Succeeds otherwise.

Compilation flags:
`static`

Template:
`print_mutation(Flag,Original,Mutation)`
Mode and number of proofs:
`print_mutation(+boolean,@callable,@callable) - one`

---

target_predicate/3

True iff `Term` is a candidate for mutation.

Compilation flags:
`static`

Template:
`target_predicate(Term,Entity,Predicate)`
Mode and number of proofs:
`target_predicate(@callable,@entity_identifier,@predicate_indicator) - one`

---

target_predicate_clause_index/4

True iff `Term` is a candidate for mutation while also returning its current 1-based contiguous clause index for the matching predicate or non-terminal.

Compilation flags:

```
static
```

Template:

```
target_predicate_clause_index(Term,Entity,Predicate,ClauseIndex)
```

Mode and number of proofs:

```
target_predicate_clause_index(@callable,@entity_identifier,@predicate_indicator,
-integer) - zero_or_one
```

---

target_scope_directive/3

True iff `Term` is a matching predicate or non-terminal scope directive candidate for mutation.

Compilation flags:

```
static
```

Template:

```
target_scope_directive(Term,Entity,Predicate)
```

Mode and number of proofs:

```
target_scope_directive(@callable,@entity_identifier,@predicate_indicator) - one
```

---

target_scope_directive_index/4

True iff `Term` is a matching predicate or non-terminal scope directive candidate for mutation while also returning its 1-based index among matching scope directives for the selected predicate or non-terminal.

Compilation flags:

```
static
```

Template:

```
target_scope_directive_index(Term,Entity,Predicate,DirectiveIndex)
```

Mode and number of proofs:

```
target_scope_directive_index(@callable,@entity_identifier,@predicate_indicator,
-integer) - zero_or_one
```

---

target_predicate_directive/3

True iff `Term` is a matching predicate or non-terminal directive candidate for mutation.

Compilation flags:
`static`

Template:
`target_predicate_directive(Term,Entity,Predicate)`
Mode and number of proofs:
`target_predicate_directive(@callable,@entity_identifier,@predicate_indicator)` - `one`

---

target_predicate_directive_index/4

True iff `Term` is a matching predicate or non-terminal directive candidate for mutation while also returning its 1-based index among matching directives for the selected predicate or non-terminal.

Compilation flags:
`static`

Template:
`target_predicate_directive_index(Term,Entity,Predicate,DirectiveIndex)`
Mode and number of proofs:
`target_predicate_directive_index(@callable,@entity_identifier,@predicate_indicator,-integer)` - `zero_or_one`

---

target_uses_directive/3

True iff `Term` is a matching `uses/2` directive candidate for mutation.

Compilation flags:
`static`

Template:
`target_uses_directive(Term,Entity,Predicate)`
Mode and number of proofs:
`target_uses_directive(@callable,@entity_identifier,@predicate_indicator)` - `one`

---

target_uses_directive_index/4

True iff `Term` is a matching `uses/2` directive candidate for mutation while also returning its 1-based index among matching directives for the selected predicate or non-terminal.

Compilation flags:

```
static
```

Template:

```
target_uses_directive_index(Term,Entity,Predicate,DirectiveIndex)
```

Mode and number of proofs:

```
target_uses_directive_index(@callable,@entity_identifier,@predicate_indicator,
-integer) - zero_or_one
```

---

next_occurrence/1

Next mutation occurrence.

Compilation flags:

```
static
```

Template:

```
next_occurrence(Occurrence)
```

Mode and number of proofs:

```
next_occurrence(-integer) - one
```

Private predicates

current_predicate_clause_index_/2

Table of current clause indexes per predicate.

Compilation flags:

```
dynamic
```

Template:

```
current_predicate_clause_index_(Predicate,ClauseIndex)
```

Mode and number of proofs:

```
current_predicate_clause_index_(?predicate_indicator,?integer) - zero_or_one
```

---

update_target_predicate_clause_index_/2

Updates and returns the next clause index for the given predicate.

Compilation flags:
`static`

Template:
`update_target_predicate_clause_index_(Predicate,ClauseIndex)`
Mode and number of proofs:
`update_target_predicate_clause_index_(@predicate_indicator,-integer)` - `one`

---

current_scope_directive_index_/2

Table of current scope directive indexes per predicate.

Compilation flags:
`dynamic`

Template:
`current_scope_directive_index_(Predicate,DirectiveIndex)`
Mode and number of proofs:
`current_scope_directive_index_(?predicate_indicator,?integer)` - `zero_or_one`

---

update_target_scope_directive_index_/2

Updates and returns the next scope directive index for the given predicate.

Compilation flags:
`static`

Template:
`update_target_scope_directive_index_(Predicate,DirectiveIndex)`
Mode and number of proofs:
`update_target_scope_directive_index_(@predicate_indicator,-integer)` - `one`

---

current_predicate_directive_index_/2

Table of current predicate directive indexes per predicate.

Compilation flags:
`dynamic`

Template:
`current_predicate_directive_index_(Predicate,DirectiveIndex)`
Mode and number of proofs:
`current_predicate_directive_index_(?predicate_indicator,?integer)` - `zero_or_one`

---

update_target_predicate_directive_index_/2

Updates and returns the next predicate directive index for the given predicate.

Compilation flags:
`static`

Template:
`update_target_predicate_directive_index_(Predicate,DirectiveIndex)`
Mode and number of proofs:
`update_target_predicate_directive_index_(@predicate_indicator,-integer)` - `one`

---

current_uses_directive_index_/2

Table of current uses directive indexes per predicate.

Compilation flags:
`dynamic`

Template:
`current_uses_directive_index_(Predicate,DirectiveIndex)`
Mode and number of proofs:
`current_uses_directive_index_(?predicate_indicator,?integer)` - `zero_or_one`

---

update_target_uses_directive_index_/2

Updates and returns the next uses directive index for the given predicate.

Compilation flags:
`static`

Template:
`update_target_uses_directive_index_(Predicate,DirectiveIndex)`
Mode and number of proofs:
`update_target_uses_directive_index_(@predicate_indicator,-integer)` - `one`

---

seen_/1

Table of last seen mutation occurrence.

Compilation flags:
`dynamic`

Template:
`seen_(Occurrence)`
Mode and number of proofs:
`seen_(?integer)` - `zero_or_one`

Operators

(none)
protocol

1.228.12 mutator_protocol

Mutator protocol.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-07

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *reset/0*
  - *mutation/2*
  - *coverage_clause_mutator/0*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

reset/0

Resets any mutator internal state used while expanding terms.

Compilation flags:
`static`

Mode and number of proofs:
`reset` - `one`

---

mutation/2

Generates by backtracking zero or more mutations for a given term.

Compilation flags:
`static`

Template:
`mutation(Term,Mutation)`

Mode and number of proofs:
`mutation(@callable,@callable)` - `zero_or_more`

---

coverage_clause_mutator/0

True when mutation occurrences map directly to predicate clause numbers and can use baseline clause coverage for skipping uncovered mutants.

Compilation flags:
`static`

Mode and number of proofs:
`coverage_clause_mutator` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.228.13 predicate_directive_suppression(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting directives.
- `DirectiveIndex` - 1-based index for the selected matching directive.
- `Occurrence` - 1-based mutation occurrence index to target within selected matching directives.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `predicate_directive_suppression` mutator by suppressing matching predicate or non-terminal directives.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
`public` *directive_mutator_protocol*
Imports:

`public` *mutator_common*

Remarks:
(none)

Inherited public predicates:
coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.228.14 `relational_operator_inversion(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)`

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting clauses to mutate.
- `ClauseIndex` - 1-based clause index for the selected mutation.
- `Occurrence` - 1-based mutation occurrence index to target within selected predicate clauses.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `relational_operator_inversion` mutator for matching predicate clauses.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-18

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
`public` *clause_mutator_protocol*
Imports:
`public` *mutator_common*

Remarks:
(none)

Inherited public predicates:
coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.228.15 `relational_operator_replacement(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)`

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting clauses to mutate.
- `ClauseIndex` - 1-based clause index for the selected mutation.
- `Occurrence` - 1-based mutation occurrence index to target within selected predicate clauses.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `relational_operator_replacement` mutator for matching predicate clauses.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-03-20

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
`public` *clause_mutator_protocol*

Imports:
`public` *mutator_common*

Remarks:
(none)

Inherited public predicates:
coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.228.16 `scope_directive_replacement(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)`

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting directives.
- `DirectiveIndex` - 1-based index for the selected matching scope directive.
- `Occurrence` - 1-based mutation occurrence index selecting an alternative visibility for the directive.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `scope_directive_replacement` mutator by replacing matching scope directives with an alternative visibility.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
`public` *directive_mutator_protocol*
Imports:
`public` *mutator_common*

Remarks:
(none)

Inherited public predicates:
coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.228.17 subprocess_coverage_hook

Subprocess coverage hook object. Loaded in baseline coverage subprocesses to capture lgtunit coverage and persist it to a file.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-22

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *load_coverage_config/1*
- *Protected predicates*
- *Private predicates*
  - *coverage_file_/1*
  - *coverage_entry_/4*
- *Operators*

Public predicates

load_coverage_config/1

Loads a baseline coverage config file and enables coverage collection in the subprocess.

Compilation flags:
`static`

Template:
`load_coverage_config(ConfigFile)`
Mode and number of proofs:
`load_coverage_config(+atom)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

coverage_file_/1

Holds the baseline coverage output file path for the current subprocess.

Compilation flags:
`dynamic`

Template:
`coverage_file_(CoverageFile)`
Mode and number of proofs:
`coverage_file_(?atom)` - `zero_or_more`

---

coverage_entry_/4

Captured lgtunit coverage for one entity predicate: covered clauses and total clauses.

Compilation flags:
`dynamic`

Template:
`coverage_entry_(Entity,Predicate,CoveredClauses,TotalClauses)`
Mode and number of proofs:
`coverage_entry_(?entity_identifier,?predicate_indicator,?list(integer),?integer)` - `zero_or_more`

Operators

(none)
object

1.228.18 subprocess_mutation_hook

Subprocess mutation hook object. Loaded in mutation testing subprocesses to set up the mutator hook and record test results to a status file.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-10-22

Compilation flags:
`static, context_switching_calls`

Provides:
logtalk::message_hook/4
Uses:
fast_random(Algorithm)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *load_config/1*
- *Protected predicates*
- *Private predicates*
  - *status_file_/1*
  - *failed_count_/1*
- *Operators*

Public predicates

load_config/1

Loads a mutation config file and sets up the mutator hook for the subprocess.

Compilation flags:
`static`

Template:
`load_config(ConfigFile)`
Mode and number of proofs:
`load_config(+atom)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

status_file_/1

Holds the status file path for the current subprocess.

Compilation flags:
`dynamic`

Template:
`status_file_(StatusFile)`
Mode and number of proofs:
`status_file_(?atom)` - `zero_or_more`

---

failed_count_/1

Accumulates the number of failed tests reported by lgtunit.

Compilation flags:
`dynamic`

Template:
`failed_count_(Failed)`
Mode and number of proofs:
`failed_count_(?integer)` - `zero_or_more`

Operators

(none)
object

1.228.19 truth_literal_flip(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting clauses to mutate.
- `ClauseIndex` - 1-based clause index for the selected mutation (equal to `Occurrence` for this mutator).
- `Occurrence` - 1-based mutation occurrence index to target within selected predicate clauses.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `truth_literal_flip` mutator for matching predicate clauses.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-03-20

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
`public` *clause_mutator_protocol*

Imports:
`public` *mutator_common*

Remarks:
(none)

Inherited public predicates:
coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.228.20 `uses_directive_resource_deletion(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)`

- `Entity` - Identifier of the entity being mutated.
- `Predicate` - Predicate or non-terminal indicator selecting resources.
- `DirectiveIndex` - 1-based index for the selected matching `uses/2` directive.
- `Occurrence` - 1-based mutation occurrence index selecting which matching resource to delete.
- `PrintMutation` - Boolean flag to print the original and mutated term plus source location.

Hook object implementing the `uses_directive_resource_deletion` mutator by deleting one matching resource from a `uses/2` directive.

Availability:
`logtalk_load(mutation_testing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-03-20

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
`public` *directive_mutator_protocol*
Imports:
`public` *mutator_common*

Remarks:
(none)

Inherited public predicates:
coverage_clause_mutator/0 *goal_expansion/2* *mutation/2* *reset/0* *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.229 mutations

object

1.229.1 `default_atom_mutations`

Default atom mutations.

Availability:
`logtalk_load(mutations(loader))`

Author: Paulo Moura
Version: 0:1:0
Date: 2023-11-24

Compilation flags:
`static, context_switching_calls`

Provides:
mutations_store::mutation/4
Uses:
fast_random
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

type

object

1.229.2 `default_compound_mutations`

Default compound mutations.

Availability:
`logtalk_load(mutations(loader))`

Author: Paulo Moura
Version: 0:1:0
Date: 2023-11-23

Compilation flags:
`static, context_switching_calls`

Provides:
mutations_store::mutation/4
Uses:
mutations_store

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

type

object

1.229.3 `default_float_mutations`

Default float mutations.

Availability:
`logtalk_load(mutations(loader))`

Author: Paulo Moura
Version: 0:1:0
Date: 2023-11-23

Compilation flags:
`static, context_switching_calls`

Provides:
mutations_store::mutation/4

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

type

object

1.229.4 `default_integer_mutations`

Default integer mutations.

Availability:
`logtalk_load(mutations(loader))`

Author: Paulo Moura
Version: 0:1:0
Date: 2023-11-24

Compilation flags:
`static, context_switching_calls`

Provides:
mutations_store::mutation/4
Uses:
fast_random
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

type

object

1.229.5 `default_list_mutations`

Default list mutations.

Availability:
`logtalk_load(mutations(loader))`

Author: Paulo Moura
Version: 0:1:0
Date: 2023-11-24

Compilation flags:
`static, context_switching_calls`

Provides:
mutations_store::mutation/4
Uses:
fast_random
list
mutations_store

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

type

category

1.229.6 `mutations`

Adds mutations support to the library `type` object.

Availability:
`logtalk_load(mutations(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2023-11-23

Compilation flags:
`static`

Complements:
type
Uses:
mutations_store

Remarks:

(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *mutation/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

mutation/3

Returns a random mutation of a term into another term of the same type. The input `Term` is assume to be valid for the given `Type`.

Compilation flags:
`static`

Template:
`mutation(Type,Term,Mutation)`
Mode and number of proofs:
`mutation(@callable,@term,-term) - one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.229.7 mutations_store

Stores mutation definitions for selected types. User extensible by defining objects or categories defining clauses for the `mutation/3` predicate and using this object as a hook object for their compilation.

Availability:
`logtalk_load(mutations(loader))`

Author: Paulo Moura

Version: 0:1:0
Date: 2023-11-23

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
fast_random

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
  - *mutation/3*
  - *counter/2*
- *Protected predicates*
- *Private predicates*
  - *mutation/4*
  - *counter_/2*
- *Operators*

Public predicates

mutation/3

Returns a random mutation of a term into another term of the same type. The input `Term` is assumed to be valid for the given `Type`.

Compilation flags:
`static`

Template:
`mutation(Type,Term,Mutation)`
Mode and number of proofs:
`mutation(@callable,@term,-term) - one`

counter/2

Table of the number of mutations available per type.

Compilation flags:
`static`

Template:
`counter(Type,N)`
Mode and number of proofs:
`counter(?callable,?positive_integer)` - `zero_or_more`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

mutation/4

Returns a random mutation of a term into another term of the same type using mutator `N`. The input `Term` is assume to be valid for the given `Type`.

Compilation flags:
`static, multifile`

Template:
`mutation(Type,N,Term,Mutation)`
Mode and number of proofs:
`mutation(?callable,?positive_integer,@term,-term)` - `zero_or_more`

---

counter_/2

Internal counter for the number of mutations available for a given type.

Compilation flags:
`dynamic`

Template:
`counter_(Type,N)`
Mode and number of proofs:
`counter_(?callable,?positive_integer)` - `zero_or_more`

Operators

(none)

See also

type

1.230 mutual_information_feature_selector

object

1.230.1 `mutual_information_feature_selector`

Selects categorical or discretized continuous features using mutual information or symmetrical uncertainty.

Availability:
`logtalk_load(mutual_information_feature_selector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *filter_feature_selector_common*
`public` *feature_discretization*

Remarks:
(none)

Inherited public predicates:
check_option/1 check_options/1 check_selector/1 default_option/1 default_options/1 diagnostic/2 diagnostics/2 export_to_clauses/4 export_to_file/4 feature_scores/2 learn/2 learn/3 option/2 option/3 print_selector/1 selected_features/2 selector_options/2 valid_option/1 valid_options/1 valid_selector/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *score_columns/3*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`score_columns/3`

Scores prepared columns using sparse contingency counts.

Compilation flags:
`static`

Template:
`score_columns(Columns,Criterion,Scores)`
Mode and number of proofs:
`score_columns(+list(compound),+atom,-list(pair))` - `one`

Operators

(none)

1.231 n_grams

object

1.231.1 `n_grams(Representation)`

- `Representation` - Text representation. Valid values are `atom`, `codes`, and `chars`.

Token and character n-gram generation for multiple text representations.

Availability:
`logtalk_load(n_grams(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-05

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *n_grams_protocol*
Imports:
`public` *options*

Uses:

list
type

Remarks:

(none)

Inherited public predicates:

bigrams/2 *character_n_grams/3* *character_n_grams/4* *check_option/1* *check_options/1* *count/2* *count/3* *default_option/1* *default_options/1* *n_grams/3* *n_grams/4* *option/2* *option/3* *trigrams/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

n_grams_protocol

protocol

1.231.2 `n_grams_protocol`

N-gram generation and counting protocol.

Availability:

`logtalk_load(n_grams(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-11

Compilation flags:

`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *n_grams/3*
  - *n_grams/4*
  - *character_n_grams/3*
  - *character_n_grams/4*
  - *count/2*
  - *count/3*
  - *bigrams/2*
  - *trigrams/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

n_grams/3

Generates overlapping n-grams from a list of text tokens.

Compilation flags:
`static`

Template:
`n_grams(N,Tokens,NGrams)`

Mode and number of proofs:
`n_grams(+positive_integer,+list(text),-list(list(text)))` - `one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
`N` is a variable:
`instantiation_error`
`N` is neither a variable nor a positive integer:
`type_error(positive_integer,Options)`

`Tokens` is a partial list or a list with an element `Element` which is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but a ground element `Element` of the list `Tokens` is not an atom:
`type_error(atom,Element)`
The `Representation` parameter is `chars` but an element `Element` of the list `Tokens` is not a list of characters:
`type_error(chars,Element)`
The `Representation` parameter is `codes` but an element `Element` of the list `Tokens` is not a list of character codes:
`type_error(codes,Element)`

---

n_grams/4

Generates n-grams from a list of text tokens using the given options.

Compilation flags:
`static`

Template:
`n_grams(N,Tokens,NGrams,Options)`
Mode and number of proofs:
`n_grams(+positive_integer,+list(text),-list(list(text)),+list(compound))` - `one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
`N` is a variable:
`instantiation_error`
`N` is neither a variable nor a positive integer:
`type_error(positive_integer,Options)`
`Tokens` is a partial list or a list with an element `Element` which is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but a ground element `Element` of the list `Tokens` is not an atom:
`type_error(atom,Element)`
The `Representation` parameter is `chars` but an element `Element` of the list `Tokens` is not a list of characters:
`type_error(chars,Element)`
The `Representation` parameter is `codes` but an element `Element` of the list `Tokens` is not a list of character codes:
`type_error(codes,Element)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

character_n_grams/3

Generates overlapping character n-grams from a text value.

Compilation flags:
`static`

Template:
`character_n_grams(N,Text,NGrams)`
Mode and number of proofs:
`character_n_grams(+positive_integer,+text,-list(text))` - `one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
`N` is a variable:
`instantiation_error`
`N` is neither a variable nor a positive integer:
`type_error(positive_integer,Options)`
`Text` is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but `Text` is not an atom:
`type_error(atom,Text)`
The `Representation` parameter is `chars` but `Text` is not a list of characters:
`type_error(chars,Text)`
The `Representation` parameter is `codes` but `Text` is not a list of character codes:
`type_error(codes,Text)`

---

character_n_grams/4

Generates character n-grams from a text value using the given options.

Compilation flags:
`static`

Template:
`character_n_grams(N,Text,NGrams,Options)`
Mode and number of proofs:
`character_n_grams(+positive_integer,+text,-list(text),+list(compound))` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

```
instantiation_error
```

The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:

```
domain_error(text_representation,Representation)
```

`N` is a variable:

```
instantiation_error
```

`N` is neither a variable nor a positive integer:

```
type_error(positive_integer,Options)
```

`Text` is not a ground term:

```
instantiation_error
```

The `Representation` parameter is `atom` but `Text` is not an atom:

```
type_error(atom,Text)
```

The `Representation` parameter is `chars` but `Text` is not a list of characters:

```
type_error(chars,Text)
```

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

```
type_error(codes,Text)
```

`Options` is a variable:

```
instantiation_error
```

`Options` is neither a variable nor a list:

```
type_error(list,Options)
```

An element `Option` of the list `Options` is a variable:

```
instantiation_error
```

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

---

count/2

Counts n-gram occurrences, preserving the order of first occurrence. Assumes `NGrams` is valid.

Compilation flags:

```
static
```

Template:

```
count(NGrams,Counts)
```

Mode and number of proofs:

```
count(+list(text),-list(pair(text,positive_integer))) - one
count(+list(list(text)),-list(pair(list(text),positive_integer))) - one
```

---

count/3

Counts n-gram occurrences using the requested ordering. Valid orderings are `first_occurrence`, `standard`, and `frequency_descending`. Assumes `NGrams` is valid.

Compilation flags:

`static`

Template:

`count(Order,NGrams,Counts)`

Mode and number of proofs:

`count(+atom,+list(text),-list(pair(text,positive_integer)))` - `one_or_error`

`count(+atom,+list(list(text)),-list(pair(list(text),positive_integer)))` - `one_or_error`

Exceptions:

`Order` is a variable:

`instantiation_error`

`Order` is neither a variable nor a valid ordering:

`domain_error(count_order,Order)`

---

bigrams/2

Generates overlapping bigrams from a list of text tokens.

Compilation flags:

`static`

Template:

`bigrams(Tokens,Bigrams)`

Mode and number of proofs:

`bigrams(+list(text),-list(list(text)))` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

`Tokens` is a partial list or a list with an element `Element` which is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but a ground element `Element` of the list `Tokens` is not an atom:

`type_error(atom,Element)`

The `Representation` parameter is `chars` but an element `Element` of the list `Tokens` is not a list of characters:

`type_error(chars,Element)`

The `Representation` parameter is `codes` but an element `Element` of the list `Tokens` is not a list of character codes:

`type_error(codes,Element)`

`trigrams/2`

Generates overlapping trigrams from a list of text tokens.

Compilation flags:
`static`

Template:
`trigrams(Tokens,Trigrams)`

Mode and number of proofs:
`trigrams(+list(text),-list(list(text)))` - `one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
`Tokens` is a partial list or a list with an element `Element` which is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but a ground element `Element` of the list `Tokens` is not an atom:
`type_error(atom,Element)`
The `Representation` parameter is `chars` but an element `Element` of the list `Tokens` is not a list of characters:
`type_error(chars,Element)`
The `Representation` parameter is `codes` but an element `Element` of the list `Tokens` is not a list of character codes:
`type_error(codes,Element)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

n_grams(Representation)

1.232 naive_bayes_classifier

object

1.232.1 `naive_bayes_classifier`

Naive Bayes classifier with Laplace smoothing and Gaussian distribution support. Learns from a dataset object implementing the `dataset_protocol` protocol and returns a classifier term that can be used for prediction and exported as predicate clauses.

Availability:
`logtalk_load(naive_bayes_classifier(loader))`

Author: Paulo Moura
Version: 2:0:1
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *probabilistic_classifier_common*
Uses:
format
list
pairs
population
type

Remarks:
(none)

Inherited public predicates:
check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *isolation_forest_anomaly_detector*, *c45_classifier*, *knn_classifier*, *nearest_centroid_classifier*, *random_forest_classifier*, *adaptive_boosting_classifier*

1.233 nanoid

object

1.233.1 `nanoid`

NanoID generator using atom representation, 21 symbols, and the standard URL alphabet.

Availability:
`logtalk_load(nanoid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *nanoid(atom,21,_-0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ)*

Remarks:
(none)

Inherited public predicates:
generate/1

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

nanoid(Representation,Size,Alphabet), *cuid2*, *ids*, *ksuid*, *snowflakeid*, *sqids*, *typeid*, *ulid*, *uuid*

object

1.233.2 `nanoid(Representation,Size,Alphabet)`

- `Representation` - Text representation for the NanoID. Possible values are `atom`, `chars`, and `codes`.
- `Size` - Number of symbols in the NanoID.
- `Alphabet` - Alphabet used for generating NanoIDs represented as an atom, list of characters, or list of character codes.

NanoID generator.

Availability:
`logtalk_load(nanoid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *nanoid_protocol*
Uses:
crypto
list

Remarks:

(none)

Inherited public predicates:

generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

nanoid, *cuid2(Representation,Size,Alphabet)*, *ids(Representation,Bytes)*, *ksuid(Representation,Alphabet)*, *snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node)*, *sqids(Representation)*, *typeid(Representation)*, *ulid(Representation)*, *uuid(Representation)*

protocol

1.233.3 `nanoid_protocol`

NanoID generator protocol.

Availability:

```
logtalk_load(nanoid(loader))
```

Author: Paulo Moura
Version: 0:1:0
Date: 2026-02-26

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *generate/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

generate/1

Returns a NanoID.

Compilation flags:
`static`

Template:
`generate(NanoID)`
Mode and number of proofs:
`generate(--ground)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.234 nearest_centroid_classifier

object

1.234.1 nearest_centroid_classifier

Nearest Centroid classifier with multiple distance metrics. Learns from a dataset object implementing the `dataset_protocol` protocol and returns a classifier term that can be used for prediction and exported as predicate clauses.

Availability:
`logtalk_load(nearest_centroid_classifier(loader))`

Author: Paulo Moura
Version: 2:0:1
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *probabilistic_classifier_common*

Uses:
format
linear_algebra
list
numberlist
pairs
type

Remarks:
(none)

Inherited public predicates:
check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *predict/4*
  - *predict_probabilities/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

predict/4

Predicts the class label for a new instance using the learned classifier and the given options. The instance is a list of `Attribute-Value` pairs.

Compilation flags:

```
static
```

Template:

```
predict(Classifier,Instance,Class,Options)
```

Mode and number of proofs:

```
predict(+compound,+list,-atom,+list(compound)) - one
```

---

predict_probabilities/4

Predicts class probabilities for a new instance using the learned classifier and the given options. Returns a list of `Class-Probability` pairs. The instance is a list of `Attribute-Value` pairs.

Compilation flags:

```
static
```

Template:

```
predict_probabilities(Classifier,Instance,Probabilities,Options)
```

Mode and number of proofs:

```
predict_probabilities(+compound,+list,-list,+list(compound)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *isolation_forest_anomaly_detector*, *c45_classifier*, *knn_classifier*, *naive_bayes_classifier*, *random_forest_classifier*, *adaptive_boosting_classifier*

1.235 nested_dictionaries

object

1.235.1 navltree

Nested dictionary implementation based on the AVL tree implementation. Uses standard order to compare keys.

Availability:
`logtalk_load(nested_dictionaries(loader))`

Author: Paul Brown and Paulo Moura.
Version: 0:1:0
Date: 2021-04-09

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *nested_dictionary_protocol*
Extends:
`private` *avltree*

Remarks:
(none)

Inherited public predicates:
as_curly_bracketed/2 *as_nested_dictionary/2* *delete_in/4* *empty/1* *insert_in/4* *lookup_in/3* *new/1* *update_in/4* *update_in/5*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

nrbtree, *nbintree*

object

1.235.2 `nbintree`

Nested dictionary implementation based on the simple binary tree implementation. Uses standard order to compare keys.

Availability:
`logtalk_load(nested_dictionaries(loader))`

Author: Paul Brown and Paulo Moura.
Version: 0:1:0
Date: 2021-04-09

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *nested_dictionary_protocol*
Extends:
`private` *bintree*

Remarks:
(none)

Inherited public predicates:
as_curly_bracketed/2 *as_nested_dictionary/2* *delete_in/4* *empty/1* *insert_in/4* *lookup_in/3* *new/1* *update_in/4* *update_in/5*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

nrbtree, *navltree*

protocol

1.235.3 `nested_dictionary_protocol`

Nested dictionary protocol.

Availability:
`logtalk_load(nested_dictionaries(loader))`

Author: Paul Brown and Paulo Moura
Version: 0:1:1
Date: 2026-06-14

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *new/1*
  - *empty/1*
  - *as_nested_dictionary/2*
  - *as_curly_bracketed/2*

      - *lookup_in/3*
      - *update_in/4*
      - *update_in/5*
      - *insert_in/4*
      - *delete_in/4*
    - *Protected predicates*
    - *Private predicates*
    - *Operators*

Public predicates

new/1

Create an empty (nested) dictionary.

Compilation flags:
`static`

Template:
`new(Dictionary)`
Mode and number of proofs:
`new(--dictionary)` - `one`

---

empty/1

True iff the dictionary is empty.

Compilation flags:
`static`

Template:
`empty(Dictionary)`
Mode and number of proofs:
`empty(@dictionary)` - `zero_or_one`

---

as_nested_dictionary/2

Creates a (nested) dictionary term from a curly-brackted term representation.

Compilation flags:
`static`

Template:

```
as_nested_dictionary(Term,Dictionary)
```

Mode and number of proofs:

```
as_nested_dictionary(++term,--dictionary) - one_or_error
```

Exceptions:

`Term` is a variable:

```
instantiation_error
```

`Term` is neither a variable nor a valid curly-bracketed term representation:

```
type_error(curly_bracketed_term,Term)
```

`Dictionary` is instantiated:

```
uninstantiation_error(Dictionary)
```

---

as_curly_bracketed/2

Creates a a curly-brackted term representation from a (nested) dictionary.

Compilation flags:

```
static
```

Template:

```
as_curly_bracketed(Dictionary,Term)
```

Mode and number of proofs:

```
as_curly_bracketed(+dictionary,--term) - one_or_error
```

Exceptions:

`Dictionary` is a variable or contains a variable leaf value that cannot be represented:

```
instantiation_error
```

`Term` is instantiated:

```
uninstantiation_error(Term)
```

---

lookup_in/3

Lookup a chain of keys in a nested dictionary. Unifies `Value` with `Dictionary` when `Keys` is the empty list.

Compilation flags:

```
static
```

Template:

```
lookup_in(Keys,Value,Dictionary)
```

Mode and number of proofs:

```
lookup_in(++list(ground),?term,+dictionary) - zero_or_more
```

---

update_in/4

Updates the value found by traversing through the nested keys.

Compilation flags:

```
static
```

Template:

```
update_in(OldDictionary,Keys,Value,NewDictionary)
```

Mode and number of proofs:

```
update_in(+dictionary,++list(ground),++term,--dictionary) - zero_or_one
```

---

update_in/5

Updates the value found by traversing through the nested keys, only succeeding if the value found after traversal matches the old value.

Compilation flags:

```
static
```

Template:

```
update_in(OldDictionary,Keys,OldValue,NewValue,NewDictionary)
```

Mode and number of proofs:

```
update_in(+dictionary,++list(ground),?term,++term,--dictionary) - zero_or_one
```

---

insert_in/4

Inserts a key-value pair into a dictionary by traversing through the nested keys. When the key already exists, the associated value is updated.

Compilation flags:

```
static
```

Template:

```
insert_in(OldDictionary,Keys,Value,NewDictionary)
```

Mode and number of proofs:

```
insert_in(+dictionary,++list(ground),++term,--dictionary) - zero_or_one
```

---

delete_in/4

Deletes a matching key-value pair from a dictionary by traversing through the nested keys, returning the updated dictionary.

Compilation flags:
`static`

Template:
`delete_in(OldDictionary,Keys,Value,NewDictionary)`
Mode and number of proofs:
`delete_in(+dictionary,++list(ground),?term,--dictionary)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

navltree, *nbintree*, *nrbtree*

object

1.235.4 nrbtree

Nested dictionary implementation based on the red-black tree implementation. Uses standard order to compare keys.

Availability:
`logtalk_load(nested_dictionaries(loader))`

Author: Paul Brown and Paulo Moura.
Version: 0:1:0
Date: 2021-04-09

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *nested_dictionary_protocol*
Extends:
`private` *rbtree*

Remarks:

(none)

Inherited public predicates:

as_curly_bracketed/2 *as_nested_dictionary/2* *delete_in/4* *empty/1* *insert_in/4* *lookup_in/3* *new/1* *update_in/4* *update_in/5*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

navltree, *nbintree*

1.236 nmea

object

1.236.1 `nmea`

Parser for NMEA 0183 sentences with typed semantic decoding for selected GPS/GNSS sentence types.

Availability:

```
logtalk_load(nmea(loader))
```

Author: Paulo Moura
Version: 2:1:0
Date: 2026-09-25

Compilation flags:

```
static, context_switching_calls
```

Implements:
`public` *nmea_protocol*
Imports:
`public` *options*
Uses:
date
list
reader
time

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *checksum/2* *data/2* *default_option/1* *default_options/1* *fields/2* *option/2* *option/3* *parse/2* *parse/3* *sentence_type/2* *talker/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.236.2 `nmea_protocol`

Protocol for parsing NMEA 0183 sentences and projecting supported sentence types into typed semantic terms.

Availability:
`logtalk_load(nmea(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-07-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *parse/3*
  - *talker/2*
  - *sentence_type/2*
  - *fields/2*
  - *checksum/2*
  - *data/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses NMEA sentences from a source specification into canonical raw sentence terms. Supported source specifications are `atom(Atom)`, `chars(List)`, `codes(List)`, `stream(Stream)`, and `file(Path)`.

Compilation flags:
`static`

Template:
`parse(Source,Sentences)`

Mode and number of proofs:
`parse(++compound,--list(compound))` - `one_or_error`

Exceptions:
`Source` is a variable or contains a variable source argument:
`instantiation_error`
`Source` is neither a variable nor a valid source specification:
`domain_error(nmea_source,Source)`
`Source` is a valid source specification but contains a line that cannot be parsed as a sentence:
`domain_error(nmea_sentence,Line)`

parse/3

Parses NMEA sentences from a source specification using parser options. Supported options are `checksum(required)`, `checksum(optional)`, `checksum(ignore)`, `unknown_type(keep)`, and `unknown_type(error)`.

Compilation flags:

`static`

Template:

`parse(Source,Sentences,Options)`

Mode and number of proofs:

`parse(++compound,--list(compound),++list(compound))` - `one_or_error`

Exceptions:

`Source` is a variable or contains a variable source argument:

`instantiation_error`

`Source` is neither a variable nor a valid source specification:

`domain_error(nmea_source,Source)`

`Source` is a valid source specification but contains a line that cannot be parsed as a sentence:

`domain_error(nmea_sentence,Line)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

---

talker/2

Returns the normalized talker identifier for a parsed sentence. Proprietary sentences use the atom `proprietary`.

Compilation flags:

`static`

Template:

`talker(Sentence,Talker)`

Mode and number of proofs:

`talker(+compound,-atom)` - `one`

---

sentence_type/2

Returns the normalized sentence type identifier for a parsed sentence.

Compilation flags:
`static`

Template:
`sentence_type(Sentence,Type)`
Mode and number of proofs:
`sentence_type(+compound,-atom)` - `one`

---

fields/2

Returns the lossless ordered list of sentence fields after the sentence identifier.

Compilation flags:
`static`

Template:
`fields(Sentence,Fields)`
Mode and number of proofs:
`fields(+compound,-list(atom))` - `one`

---

checksum/2

Returns the checksum information as `checksum(Provided,Computed)` where `Provided` is either a normalized two-digit hexadecimal atom or `missing`.

Compilation flags:
`static`

Template:
`checksum(Sentence,Checksum)`
Mode and number of proofs:
`checksum(+compound,-compound)` - `one`

---

data/2

Projects a parsed sentence into a typed semantic term for the supported sentence types `gga`, `rmc`, `gsa`, `gsv`, `vtg`, and `gll`.

Compilation flags:
`static`

Template:
`data(Sentence,Data)`
Mode and number of proofs:
`data(+compound,-compound)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.237 nmf_projection

object

1.237.1 nmf_projection

Non-negative Matrix Factorization dimension reduction for non-negative continuous datasets.

Availability:
`logtalk_load(nmf_projection(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-18

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *dimension_reducer_common*
Uses:
format
linear_algebra
list

numberlist
type
user

Remarks:
(none)

Inherited public predicates:
check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.238 numerical_analysis

object

1.238.1 `adaptive_simpson_quadrature(Function)`

- `Function` - Object implementing `univariate_function_protocol`.

Adaptive Simpson quadrature with absolute and relative error tolerances.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *quadrature(Function)*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *integrate/3* *integrate/4* *integrate/5* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

quadrature_protocol, *univariate_function_protocol*

object

1.238.2 `barycentric_interpolator`

Barycentric polynomial interpolation with weights precomputed during fitting.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-09-04

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *interpolator*
`public` *options*

Uses:
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *evaluate/3* *fit/2* *fit/3* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

interpolator_protocol, *piecewise_linear_interpolator*, *cubic_spline_interpolator*

object

1.238.3 `bisection_root_finder(Function)`

- `Function` - Object implementing `univariate_function_protocol`.

Bisection root finder for a continuous scalar function over a sign-changing bracket.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *root_finder(Function)*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *find_root/2* *find_root/3* *find_root/4* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

root_finder_protocol, *univariate_function_protocol*

object

1.238.4 `brent_root_finder(Function)`

- `Function` - Object implementing `univariate_function_protocol`.

Brent-Dekker derivative-free root finder combining interpolation with bisection safeguards.

Availability:

```
logtalk_load(numerical_analysis(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-25

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *root_finder(Function)*

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *find_root/2* *find_root/3* *find_root/4* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

root_finder_protocol, *bisection_root_finder(Function)*, *secant_root_finder(Function)*

object

1.238.5 `cubic_spline_interpolator`

Natural and clamped cubic-spline interpolation using a tridiagonal solve.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *interpolator*
`public` *options*
Uses:
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *evaluate/3* *fit/2* *fit/3* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*

  - derivative/4
- Protected predicates
- Private predicates
- Operators

Public predicates

derivative/4

Evaluates the first or second derivative of a fitted cubic spline.

Compilation flags:
`static`

Template:
`derivative(Model,Argument,Order,Value)`

Mode and number of proofs:
`derivative(+compound,+number,+integer,-number)` - `one_or_error`

Exceptions:
`Order` is not one or two:
`domain_error(derivative_order,Order)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

interpolator_protocol, *piecewise_linear_interpolator*, *barycentric_interpolator*

object

1.238.6 `euler_ode_solver(System)`

- `System` - Object implementing `ode_system_protocol`.

Fixed-step explicit Euler solver for non-stiff initial-value problems.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ode_solver(System)*

Uses:
linear_algebra
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *solve/4* *solve/5* *solve/6* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ode_solver_protocol, *rk4_ode_solver(System)*, *rk45_ode_solver(System)*

object

1.238.7 gauss_legendre_quadrature(Function)

- `Function` - Object implementing `univariate_function_protocol`.

Fixed-order Gauss-Legendre quadrature for orders 2, 4, 8, and 16.

Availability:

```
logtalk_load(numerical_analysis(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *quadrature(Function)*

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *integrate/3* *integrate/4* *integrate/5* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

quadrature_protocol, *adaptive_simpson_quadrature(Function)*

category

1.238.8 `interpolator`

Common point validation and domain handling for one-dimensional interpolators.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static`

Implements:
`public` *interpolator_protocol*
Uses:
list

Remarks:
(none)

Inherited public predicates:
evaluate/3 *fit/2* *fit/3*

- *Public predicates*
- *Protected predicates*
  - *prepare_points/3*

  - *check_argument/3*
  - *select_interval/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

prepare_points/3

Validates and sorts interpolation points and returns separate abscissa and ordinate lists.

Compilation flags:
`static`

Template:
`prepare_points(Points,Abscissas,Ordinates)`
Mode and number of proofs:
`prepare_points(+list(pair),-list(number),-list(number))` - `one_or_error`

Exceptions:
`Points` is a variable:
`instantiation_error`
`Points` is not a valid list of at least two numeric points:
`domain_error(interpolation_points,Points)`
`Points` contains a duplicate abscissa:
`domain_error(duplicate_abscissa,Abscissa)`

---

check_argument/3

Checks that an interpolation argument lies in the closed fitted domain.

Compilation flags:
`static`

Template:
`check_argument(Argument,Lower,Upper)`
Mode and number of proofs:
`check_argument(+number,+number,+number)` - `one_or_error`

Exceptions:
`Argument` is a variable:
`instantiation_error`

`Argument` is not a number:
`type_error(number,Argument)`
`Argument` lies outside the fitted domain:
`domain_error(interpolation_domain,Argument)`

`select_interval/3`

Selects the fitted interval containing an already validated argument.

Compilation flags:
`static`

Template:
`select_interval(Argument,Intervals,Interval)`
Mode and number of proofs:
`select_interval(+number,+list(compound),-compound)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.238.9 `interpolator_protocol`

Protocol for fitting and evaluating one-dimensional interpolants.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *fit/2*
  - *fit/3*
  - *evaluate/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

fit/2

Fits an interpolant to a list of `X-Y` points using default options.

Compilation flags:
`static`

Template:
`fit(Points,Model)`
Mode and number of proofs:
`fit(+list(pair),-compound)` - `one_or_error`

Exceptions:
`Points` is a variable:
`instantiation_error`
`Points` is not a valid list of at least two numeric points with distinct abscissas:
`domain_error(interpolation_points,Points)`

---

fit/3

Fits an interpolant to a list of `X-Y` points using the given options.

Compilation flags:
`static`

Template:
`fit(Points,Model,Options)`
Mode and number of proofs:
`fit(+list(pair),-compound,+list(compound))` - `one_or_error`

Exceptions:
`Points` or `Options` is a variable:

`instantiation_error`

`Points` is not a valid list of at least two numeric points with distinct abscissas:

`domain_error(interpolation_points,Points)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

---

evaluate/3

Evaluates a fitted interpolant inside its data domain.

Compilation flags:

`static`

Template:

`evaluate(Model,Argument,Value)`

Mode and number of proofs:

`evaluate(+compound,+number,-number)` - `one_or_error`

Exceptions:

`Model` is a variable:

`instantiation_error`

`Model` is not a model for the receiving interpolator:

`domain_error(interpolation_model,Model)`

`Argument` is a variable:

`instantiation_error`

`Argument` is not a number:

`type_error(number,Argument)`

`Argument` lies outside the fitted domain:

`domain_error(interpolation_domain,Argument)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.238.10 `newton_root_finder(Function)`

- `Function` - Object implementing `univariate_function_protocol` and defining `derivative/2`.

Newton root finder using a function first derivative and one initial guess.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-25

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *root_finder(Function)*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *find_root/2* *find_root/3* *find_root/4* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

root_finder_protocol, *bisection_root_finder(Function)*, *secant_root_finder(Function)*

category

1.238.11 `ode_solver(System)`

- `System` - Object implementing `ode_system_protocol`.

Common validation and option handling for non-stiff initial-value ODE solvers.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static`

Implements:
`public` *ode_solver_protocol*
Extends:
`public` *options*
Uses:
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *solve/4* *solve/5* *solve/6* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *check_initial_value/3*
  - *evaluate_derivative/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

check_initial_value/3

Checks initial and final times and a non-empty numeric initial state.

Compilation flags:
`static`

Template:
`check_initial_value(InitialTime,InitialState,FinalTime)`
Mode and number of proofs:
`check_initial_value(+number,+list(number),+number) - one_or_error`

Exceptions:
An argument is a variable:
`instantiation_error`
A time is not a number:
`type_error(number,Time)`
`InitialState` is not a non-empty numeric list:
`domain_error(ode_state,InitialState)`

---

evaluate_derivative/3

Evaluates and validates a system derivative.

Compilation flags:
`static`

Template:
`evaluate_derivative(Time,State,Derivative)`
Mode and number of proofs:
`evaluate_derivative(+number,+list(number),-list(number)) - one_or_error`

Exceptions:

The system returns a non-numeric derivative or one with the wrong dimension:

```
domain_error(ode_derivative,Derivative)
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.238.12 ode_solver_protocol

Protocol for non-stiff initial-value ODE solvers.

Availability:

```
logtalk_load(numerical_analysis(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *solve/4*
  - *solve/5*
  - *solve/6*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

solve/4

Solves an initial-value problem using default options.

Compilation flags:
`static`

Template:
`solve(InitialTime,InitialState,FinalTime,Trajectory)`
Mode and number of proofs:
`solve(+number,+list(number),+number,-list(pair))` - `one_or_error`

Exceptions:
An input argument is a variable:
`instantiation_error`
A time is not a number:
`type_error(number,Time)`
`InitialState` is not a non-empty numeric list:
`domain_error(ode_state,InitialState)`
The system returns an invalid derivative:
`domain_error(ode_derivative,Derivative)`

---

solve/5

Solves an initial-value problem using the given options.

Compilation flags:
`static`

Template:
`solve(InitialTime,InitialState,FinalTime,Trajectory,Options)`
Mode and number of proofs:
`solve(+number,+list(number),+number,-list(pair),+list(compound))` - `one_or_error`

Exceptions:
An input argument or `Options` is a variable:
`instantiation_error`
A time is not a number:
`type_error(number,Time)`
`InitialState` is not a non-empty numeric list:
`domain_error(ode_state,InitialState)`
The system returns an invalid derivative:
`domain_error(ode_derivative,Derivative)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:

`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

`solve/6`

Solves an initial-value problem and returns run statistics using the given options.

Compilation flags:
`static`

Template:
`solve(InitialTime,InitialState,FinalTime,Trajectory,Statistics,Options)`
Mode and number of proofs:
`solve(+number,+list(number),+number,-list(pair),-list(compound),+list(compound))` - `one_or_error`

Exceptions:
An input argument or `Options` is a variable:
`instantiation_error`
A time is not a number:
`type_error(number,Time)`
`InitialState` is not a non-empty numeric list:
`domain_error(ode_state,InitialState)`
The system returns an invalid derivative:
`domain_error(ode_derivative,Derivative)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.238.13 `ode_system_protocol`

Protocol for a non-stiff first-order ordinary differential equation system.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *derivative/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

derivative/3

Computes the state derivative at a time and state. The derivative must have the same dimension as the state.

Compilation flags:
```
static
```

Template:
```
derivative(Time,State,Derivative)
```
Mode and number of proofs:
```
derivative(+number,+list(number),-list(number)) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.238.14 piecewise_linear_interpolator

Piecewise-linear interpolation over a sorted set of distinct abscissas.

Availability:
```
logtalk_load(numerical_analysis(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
```
static, context_switching_calls
```

Imports:
```
public interpolator
public options
```
Uses:
```
list
```

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *evaluate/3* *fit/2* *fit/3* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

interpolator_protocol, *barycentric_interpolator*, *cubic_spline_interpolator*

category

1.238.15 `quadrature(Function)`

- `Function` - Object implementing `univariate_function_protocol`.

Common option handling and validation for scalar quadrature algorithms.

Availability:

```
logtalk_load(numerical_analysis(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:

```
static
```

Implements:

```
public quadrature_protocol
```

Extends:

`public` *options*

Uses:

type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *integrate/3* *integrate/4* *integrate/5* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - `evaluate_integrand/2`
  - `check_bounds/2`
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

`evaluate_integrand/2`

Evaluates the integrand and checks that it returns a number.

Compilation flags:

`static`

Template:

`evaluate_integrand(Argument,Value)`

Mode and number of proofs:

`evaluate_integrand(+number,-number)` - `one_or_error`

Exceptions:

The integrand returns a non-numeric value:

`domain_error(function_value,Value)`

check_bounds/2

Checks that both integration bounds are instantiated numbers.

Compilation flags:
`static`

Template:
`check_bounds(Lower,Upper)`
Mode and number of proofs:
`check_bounds(+number,+number)` - `one_or_error`

Exceptions:
An integration bound is a variable:
`instantiation_error`
An integration bound is not a number:
`type_error(number,Bound)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.238.16 quadrature_protocol

Protocol for finite-interval scalar numerical quadrature algorithms.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *integrate/3*
  - *integrate/4*
  - *integrate/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

integrate/3

Integrates the function over the given finite interval using default options.

Compilation flags:
`static`

Template:
`integrate(Lower,Upper,Integral)`
Mode and number of proofs:
`integrate(+number,+number,-number)` - `one_or_error`

Exceptions:
An integration bound is a variable:
`instantiation_error`
An integration bound is not a number:
`type_error(number,Bound)`
The function returns a non-numeric value:
`domain_error(function_value,Value)`

---

integrate/4

Integrates the function over the given finite interval using the given options.

Compilation flags:
`static`

Template:
`integrate(Lower,Upper,Integral,Options)`
Mode and number of proofs:
`integrate(+number,+number,-number,+list(compound))` - `one_or_error`

Exceptions:

An integration bound or `Options` is a variable:
`instantiation_error`
An integration bound is not a number:
`type_error(number,Bound)`
`Options` is not a list or contains an invalid option:
`domain_error(options,Options)`
The function returns a non-numeric value:
`domain_error(function_value,Value)`

`integrate/5`

Integrates the function and returns run statistics using the given options.

Compilation flags:
`static`

Template:
`integrate(Lower,Upper,Integral,Statistics,Options)`
Mode and number of proofs:
`integrate(+number,+number,-number,-list(compound),+list(compound))` - `one_or_error`

Exceptions:
An integration bound or `Options` is a variable:
`instantiation_error`
An integration bound is not a number:
`type_error(number,Bound)`
`Options` is not a list or contains an invalid option:
`domain_error(options,Options)`
The function returns a non-numeric value:
`domain_error(function_value,Value)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.238.17 `rk45_ode_solver(System)`

- `System` - Object implementing `ode_system_protocol`.

Adaptive Dormand-Prince 5(4) solver with componentwise error scaling and FSAL derivative reuse.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ode_solver(System)*
Uses:
linear_algebra
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *solve/4* *solve/5* *solve/6* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ode_solver_protocol, *euler_ode_solver(System)*, *rk4_ode_solver(System)*

object

1.238.18 `rk4_ode_solver(System)`

- `System` - Object implementing `ode_system_protocol`.

Classical fixed-step fourth-order Runge-Kutta solver for non-stiff initial-value problems.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ode_solver(System)*
Uses:
linear_algebra
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *solve/4* *solve/5* *solve/6* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ode_solver_protocol, *euler_ode_solver(System)*, *rk45_ode_solver(System)*

category

1.238.19 `root_finder(Function)`

- `Function` - Object implementing `univariate_function_protocol`.

Common option handling and validation for scalar root finders.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static`

Implements:
`public` *root_finder_protocol*
Extends:
`public` *options*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *find_root/2* *find_root/3* *find_root/4* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*

- *Protected predicates*
  - *evaluate_function/2*
  - *check_bracket/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

evaluate_function/2

Evaluates the function and checks that it returns a number.

Compilation flags:
`static`

Template:
`evaluate_function(Argument,Value)`
Mode and number of proofs:
`evaluate_function(+number,-number)` - `one_or_error`

Exceptions:
The function returns a non-numeric value:
`domain_error(function_value,Value)`

---

check_bracket/3

Checks and unpacks a root bracket.

Compilation flags:
`static`

Template:
`check_bracket(Bracket,Lower,Upper)`
Mode and number of proofs:
`check_bracket(+compound,-number,-number)` - `one_or_error`

Exceptions:
`Bracket` is a variable:
`instantiation_error`
`Bracket` is not a valid increasing numeric bracket:
`domain_error(root_bracket,Bracket)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.238.20 `root_finder_protocol`

Protocol for scalar root-finding algorithms.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *find_root/2*
  - *find_root/3*
  - *find_root/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

find_root/2

Finds a root using default options.

Compilation flags:
`static`

Template:
`find_root(Initial,Root)`
Mode and number of proofs:
`find_root(+compound,-number)` - `one_or_error`

Exceptions:
`Initial` is a variable:
`instantiation_error`
`Initial` is not valid for the root finder:
`domain_error(root_initialization,Initial)`
The function returns a non-numeric value:
`domain_error(function_value,Value)`

---

find_root/3

Finds a root using the given options.

Compilation flags:
`static`

Template:
`find_root(Initial,Root,Options)`
Mode and number of proofs:
`find_root(+compound,-number,+list(compound))` - `one_or_error`

Exceptions:
`Initial` or `Options` is a variable:
`instantiation_error`
`Initial` is not valid for the root finder:
`domain_error(root_initialization,Initial)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`
The function returns a non-numeric value:
`domain_error(function_value,Value)`

`find_root/4`

Finds a root and returns run statistics using the given options.

Compilation flags:
`static`

Template:
`find_root(Initial,Root,Statistics,Options)`
Mode and number of proofs:
`find_root(+compound,-number,-list(compound),+list(compound))` - `one_or_error`

Exceptions:
`Initial` or `Options` is a variable:
`instantiation_error`
`Initial` is not valid for the root finder:
`domain_error(root_initialization,Initial)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The function returns a non-numeric value:
`domain_error(function_value,Value)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.238.21 `secant_root_finder(Function)`

- `Function` - Object implementing `univariate_function_protocol`.

Derivative-free secant root finder using two initial guesses.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-25

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *root_finder(Function)*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *find_root/2* *find_root/3* *find_root/4* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

root_finder_protocol, *bisection_root_finder(Function)*, *newton_root_finder(Function)*

protocol

1.238.22 `univariate_function_protocol`

Protocol for real-valued functions of one real variable.

Availability:
`logtalk_load(numerical_analysis(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-04

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *evaluate/2*
  - *derivative/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

evaluate/2

Evaluates the function at the given argument.

Compilation flags:
`static`

Template:
`evaluate(Argument,Value)`
Mode and number of proofs:
`evaluate(+number,-number)` - `one`

---

derivative/2

Optional. Evaluates the first derivative at the given argument.

Compilation flags:
`static`

Template:
`derivative(Argument,Derivative)`
Mode and number of proofs:
`derivative(+number,-number)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.239 omt

object

1.239.1 omt

Process-backed single-objective optimization with Z3 and OptiMathSAT.

Availability:
`logtalk_load(omt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-14

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *omt_protocol*
Imports:
`public` *options*
Uses:
list
os
smt
smt_output
smt_process
smt_program
type
user

Remarks:
(none)

Inherited public predicates:
capability/2 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *model_binding/2* *model_value/3* *objective/3* *option/2* *option/3* *print_program/1* *print_result/1* *print_results/1* *program/1* *program/2* *result_property/2* *solve/2* *solve/3* *status/2* *unsat_core_name/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.239.2 `omt_expansion`

Opt-in expansion of OMT problem objects.

Availability:
`logtalk_load(omt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-15

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
smt_program

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.239.3 `omt_problem_protocol`

Protocol for ordered optimization problem statements and origins.

Availability:
`logtalk_load(omt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-14

Compilation flags:
`static`

Extends:
`public` *smt_problem_protocol*

Remarks:
(none)

Inherited public predicates:
smt_statement/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.239.4 `omt_protocol`

Protocol for single-objective optimization modulo theories.

Availability:
`logtalk_load(omt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-14

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *solve/2*
  - *solve/3*
  - *program/1*
  - *program/2*
  - *print_program/1*
  - *status/2*
  - *model_binding/2*
  - *model_value/3*

- unsat_core_name/2
- result_property/2
- print_result/1
- print_results/1
- capability/2
- objective/3

- Protected predicates
- Private predicates
- Operators

Public predicates

solve/2

Solves an OMT source with default options and returns one result per check.

Compilation flags:
```
static
```

Template:
```
solve(Source,Results)
```
Mode and number of proofs:
```
solve(+compound,-list(compound)) - one_or_error
```

---

solve/3

Solves with solver/1, executable/1, and time_limit/1 options. OptiMathSAT rejects nonzero time limits.

Compilation flags:
```
static
```

Template:
```
solve(Source,Results,Options)
```
Mode and number of proofs:
```
solve(+compound,-list(compound),+list(compound)) - one_or_error
```

Exceptions:

An OptiMathSAT time limit is nonzero:
```
domain_error(optimathsat_time_limit,Seconds)
```
Solver output or process status is invalid:
```
omt_solver_error(Formal,Status,Diagnostics,SourceMap)
```

---

program/1

Generates and prints a source without invoking a solver.

Compilation flags:
```
static
```

Template:
```
program(Source)
```
Mode and number of proofs:
```
program(+compound) - one_or_error
```

---

program/2

Generates source character codes without invoking a solver.

Compilation flags:
```
static
```

Template:
```
program(Source,Codes)
```
Mode and number of proofs:
```
program(+compound,-list(integer)) - one_or_error
```

---

print_program/1

Prints source character codes.

Compilation flags:
```
static
```

Template:
```
print_program(Codes)
```
Mode and number of proofs:
```
print_program(+list(integer)) - one_or_error
```

---

status/2

Returns the status of a result.

Compilation flags:
`static`

Template:
`status(Result,Status)`
Mode and number of proofs:
`status(+compound,-atom)` - `one_or_error`

---

model_binding/2

Enumerates model bindings.

Compilation flags:
`static`

Template:
`model_binding(Result,Binding)`
Mode and number of proofs:
`model_binding(+compound,-pair)` - `zero_or_more`

---

model_value/3

Returns a named model value.

Compilation flags:
`static`

Template:
`model_value(Result,Name,Value)`
Mode and number of proofs:
`model_value(+compound,+atom,-term)` - `zero_or_one`

---

unsat_core_name/2

Enumerates unsatisfiable core names.

Compilation flags:
```
static
```

Template:
```
unsat_core_name(Result,Name)
```
Mode and number of proofs:
```
unsat_core_name(+compound,-atom) - zero_or_more
```

---

result_property/2

Enumerates result properties.

Compilation flags:
```
static
```

Template:
```
result_property(Result,Property)
```
Mode and number of proofs:
```
result_property(+compound,?compound) - zero_or_more
```

---

print_result/1

Prints a result and its objectives.

Compilation flags:
```
static
```

Template:
```
print_result(Result)
```
Mode and number of proofs:
```
print_result(+compound) - one_or_error
```

---

print_results/1

Prints a list of results.

Compilation flags:
`static`

Template:
`print_results(Results)`
Mode and number of proofs:
`print_results(+list(compound))` - `one_or_error`

---

capability/2

Enumerates advisory solver capabilities.

Compilation flags:
`static`

Template:
`capability(Solver,Feature)`
Mode and number of proofs:
`capability(?atom,?atom)` - `zero_or_more`

---

objective/3

Enumerates objective expressions and exact solver-reported bounds from a get-objectives response.

Compilation flags:
`static`

Template:
`objective(Result,Expression,Bound)`
Mode and number of proofs:
`objective(+compound,-term,-term)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.240 one_class_svm_anomaly_detector

object

1.240.1 `one_class_svm_anomaly_detector`

One-class support vector machine anomaly detector with linear, polynomial, and radial basis function kernels. Learns from baseline training examples selected from a dataset object implementing the `anomaly_dataset_protocol` protocol.

Availability:
`logtalk_load(one_class_svm_anomaly_detector(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-10-12

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *anomaly_detector_common*

Uses:
format
linear_algebra
list
numberlist
pairs
population
type

Remarks:
(none)

Inherited public predicates:
anomaly_detector_options/2 *check_anomaly_detector/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict/4* *print_anomaly_detector/1* *score/3* *score_all/3* *valid_anomaly_detector/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

anomaly_dataset_protocol, *anomaly_detector_protocol*, *kernel_svm_classifier*, *linear_svm_classifier*

1.241 open_ai

object

1.241.1 `open_ai`

Facade predicates for the OpenAI-compatible public non-admin API library.

Availability:
`logtalk_load(open_ai(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Uses:
open_ai_catalog
open_api

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *document/1*
  - *operation_count/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

document/1

Returns the generated OpenAPI document for the supported OpenAI-compatible endpoint surface.

Compilation flags:
`static`

Template:
`document(Document)`
Mode and number of proofs:
`document(-compound)` - `one`

---

operation_count/1

Returns the number of supported public non-admin OpenAI operations in the pinned catalog.

Compilation flags:
`static`

Template:
`operation_count(Count)`
Mode and number of proofs:
`operation_count(-integer)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.241.2 `open_ai_api`

OpenAPI provider for the public non-admin OpenAI-compatible API surface.

Availability:
`logtalk_load(open_ai(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *open_api_provider_protocol*
Uses:
open_ai_catalog

Remarks:
(none)

Inherited public predicates:
api_info/1 *operations/1* *schema/2* *security/1* *security_scheme/2* *servers/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.241.3 `open_ai_backend_protocol`

Protocol implemented by OpenAI-compatible server backends.

Availability:
`logtalk_load(open_ai(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *handle_open_ai/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

handle_open_ai/3

Handles an OpenAI operation selected by operation identifier for the normalized HTTP request and returns a REST result term accepted by the `rest` library. The predicate is expected to succeed once for both successful and application error outcomes; plain failure or error is a backend contract violation handled by the server facade as an unexpected server error.

Compilation flags:
```
static
```

Template:
```
handle_open_ai(OperationId,Request,Result)
```
Mode and number of proofs:
```
handle_open_ai(+atom,+compound,--term) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.241.4 open_ai_catalog

Generated catalog for the public non-admin OpenAI API endpoint surface.

Availability:
```
logtalk_load(open_ai(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
```
static, context_switching_calls
```

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *spec_version/1*
  - *operation_count/1*
  - *operation/8*
  - *operation_properties/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

spec_version/1

Returns the pinned OpenAI OpenAPI document version used to generate the catalog.

Compilation flags:
```
static
```

Template:
```
spec_version(Version)
```
Mode and number of proofs:
```
spec_version(-atom) - one
```

---

operation_count/1

Returns the number of public non-admin OpenAI operations in the generated catalog.

Compilation flags:
```
static
```

Template:
```
operation_count(Count)
```
Mode and number of proofs:
```
operation_count(-integer) - one
```

---

operation/8

Enumerates generated OpenAI operation descriptors.

Compilation flags:
`static`

Template:
`operation(Id,Tag,Method,Path,Summary,Parameters,RequestBody,Responses)`
Mode and number of proofs:
`operation(?atom,?atom,?atom,?atom,?atom,?list(compound),?compound,?list(compound))` - `zero_or_more`

---

operation_properties/2

Enumerates additional generated OpenAPI properties for an operation.

Compilation flags:
`static`

Template:
`operation_properties(Id,Properties)`
Mode and number of proofs:
`operation_properties(?atom,?list(compound))` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.241.5 open_ai_client

OpenAI-compatible client facade using the `http_client` library.

Availability:
`logtalk_load(open_ai(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Uses:
http_client
list
open_ai_catalog

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *request/5*
  - *operation_url/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

request/5

Calls an OpenAI operation by operation identifier. Path parameters are supplied as `Name-Value` pairs. The body is ignored for methods without a request body. Options include `base_url/1`, `api_key/1`, `organization/1`, `project/1`, and ordinary `http_client` request options.

Compilation flags:
`static`

Template:
`request(OperationId,PathParameters,Body,Response,Options)`
Mode and number of proofs:
`request(+atom,+list(pair),+compound,--compound,+list(compound))` - `one_or_error`

Exceptions:
The catalog operation method `Method` is unsupported by the client facade:
`domain_error(open_ai_operation_method,Method)`
An element `Parameter` of the list `PathParameters` is not a `Name-Value` pair of atoms:
`domain_error(open_ai_path_parameter,Parameter)`
`Options` contains an invalid OpenAI client option:
`domain_error(open_ai_client_option,Option)`
The delegated HTTP client rejects the response stream:

`domain_error(http_response_stream,Error)`

`operation_url/4`

Builds the absolute request URL for an operation and path parameters.

Compilation flags:
`static`

Template:
`operation_url(OperationId,PathParameters,URL,Options)`
Mode and number of proofs:
`operation_url(+atom,+list(pair),-atom,+list(compound))` - `one_or_error`

Exceptions:
An element `Parameter` of the list `PathParameters` is not a `Name-Value` pair of atoms:
`domain_error(open_ai_path_parameter,Parameter)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.241.6 `open_ai_event_stream`

Small Server-Sent Events helpers used by OpenAI streaming endpoints.

Availability:
`logtalk_load(open_ai(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *event/2*
  - *frame/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

event/2

Parses a single OpenAI event-stream frame atom into `event(Name, Data)`. Unknown fields are ignored.

Compilation flags:
`static`

Template:
`event(Frame,Event)`
Mode and number of proofs:
`event(+atom,-compound)` - `one_or_error`

Exceptions:
`Frame` is not an event-stream frame atom:
`domain_error(open_ai_event_stream_frame,Frame)`

---

frame/2

Generates an event-stream frame atom from `event(Name, Data)`.

Compilation flags:
`static`

Template:
`frame(Event,Frame)`
Mode and number of proofs:
`frame(+compound,-atom)` - `one_or_error`

Exceptions:

`Event` is not an event-stream event term:
`domain_error(open_ai_event_stream_event,Event)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.241.7 `open_ai_server(Backend)`

- `Backend` - Application object implementing `open_ai_backend_protocol`.

OpenAI-compatible REST server facade backed by an application object.

Availability:
`logtalk_load(open_ai(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-13

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *http_handler_protocol*
`public` *open_api_provider_protocol*
Imports:
`public` *rest*
Uses:
http_core
list
open_ai_api
open_ai_catalog

Remarks:
(none)

Inherited public predicates:
api_info/1 *binary_body/2* *created_response/4* *form_body/2* *handle/2* *json_array_body/2* *json_body/2* *json_object_body/2* *json_response/4* *json_response/5* *no_content_response/2* *operations/1* *path_parameter/3* *problem_response/6* *query_parameter/3* *request_body/2* *request_header/3* *schema/2* *security/1* *security_scheme/2* *servers/1* *text_body/2*

- *Public predicates*
- *Protected predicates*
  - *open_api_info/1*
  - *open_api_servers/1*
  - *open_api_security/1*
  - *open_api_security_scheme/2*
  - *open_api_schema/2*
  - *endpoint/5*
  - *open_api_validate_request/1*
  - *open_api_validate_response/1*
  - *dispatch_open_ai/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

open_api_info/1

Hook returning the OpenAPI information object for the server facade.

Compilation flags:
`static`

Template:
`open_api_info(Info)`
Mode and number of proofs:
`open_api_info(-compound)` - `one`

---

open_api_servers/1

Hook returning the OpenAPI server list for the local server facade.

Compilation flags:
`static`

Template:
`open_api_servers(Servers)`
Mode and number of proofs:
`open_api_servers(-list(compound))` - `one`

---

open_api_security/1

Hook returning the OpenAPI security requirements for the server facade.

Compilation flags:
`static`

Template:
`open_api_security(Security)`
Mode and number of proofs:
`open_api_security(-list(compound))` - `zero_or_one`

---

open_api_security_scheme/2

Hook enumerating OpenAPI security schemes for the server facade.

Compilation flags:
`static`

Template:
`open_api_security_scheme(Name,Scheme)`
Mode and number of proofs:
`open_api_security_scheme(?atom,?compound)` - `zero_or_more`

---

open_api_schema/2

Hook enumerating named OpenAPI schemas for the server facade.

Compilation flags:
`static`

Template:
`open_api_schema(Name,Schema)`
Mode and number of proofs:
`open_api_schema(?atom,?compound)` - `zero_or_more`

---

endpoint/5

Hook enumerating REST endpoint descriptors derived from the generated OpenAI catalog.

Compilation flags:
```
static
```

Template:
```
endpoint(Id,Method,Path,Action,Options)
```
Mode and number of proofs:
```
endpoint(?atom,?atom,?atom,?atom,?list(compound)) - zero_or_more
```

---

open_api_validate_request/1

Hook selecting operations whose requests should be validated against the OpenAPI contract.

Compilation flags:
```
static
```

Template:
```
open_api_validate_request(Id)
```
Mode and number of proofs:
```
open_api_validate_request(?atom) - zero_or_more
```

---

open_api_validate_response/1

Hook selecting operations whose responses should be validated against the OpenAPI contract.

Compilation flags:
```
static
```

Template:
```
open_api_validate_response(Id)
```
Mode and number of proofs:
```
open_api_validate_response(?atom) - zero_or_more
```

---

dispatch_open_ai/2

Dispatches a routed OpenAI request to the backend object and returns a REST result. Backend application errors should be returned as REST result terms; unexpected backend failure or error is left to the `rest` action failure handling.

Compilation flags:
`static`

Template:
`dispatch_open_ai(Request,Result)`
Mode and number of proofs:
`dispatch_open_ai(+compound,--term)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.242 open_api

object

1.242.1 open_api

OpenAPI 3.1.0 document derivation, parsing, generation, and validation built on top of the `json` and `json_schema` libraries.

Availability:
`logtalk_load(open_api(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *http_json_term_helpers*
Uses:
http_core
json
json_schema
list
url(Representation)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *document/2*
  - *parse/2*
  - *generate/2*
  - *validate_document/1*
  - *validate_document/2*
  - *operation/3*
  - *validate_request/3*
  - *validate_request/4*
  - *validate_http_request/3*
  - *validate_http_request/4*
  - *validate_response/3*
  - *validate_response/4*
  - *validate_http_response/3*
  - *validate_http_response/4*
  - *json_media_descriptor/2*
  - *json_media_descriptor/3*
  - *json_request_body_descriptor/4*
  - *json_request_body_descriptor/5*
  - *json_response_descriptor/4*
  - *json_response_descriptor/5*
  - *problem_json_schema/1*
  - *problem_response_descriptor/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

document/2

Derives an OpenAPI 3.1.0 document in the repository-standard JSON term representation from an object implementing the `open_api_provider_protocol` protocol. When the provider also implements the `application_protocol` protocol, missing `description` and `license` info fields may be backfilled from application metadata and `homepage/1` may be mapped to the top-level `externalDocs` object. Top-level security defaults declared by `security/1` are emitted under the root `security` field. Reusable security schemes declared by `security_scheme/2` are emitted under `components.securitySchemes` and referenced scheme names and OAuth flow scopes are validated against those declarations when locally available.

Compilation flags:
`static`

Template:
`document(Provider,Document)`
Mode and number of proofs:
`document(+object_identifier,-compound)` - `one`

---

parse/2

Parses an OpenAPI document from the given source (`file(Path)`, `stream(Stream)`, `codes(List)`, `chars(List)`, or `atom(Atom)`) into the JSON term representation used by the `json` library.

Compilation flags:
`static`

Template:
`parse(Source,Document)`
Mode and number of proofs:
`parse(++compound,--term)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid JSON source term or does not contain parsable JSON by the implementation:
`domain_error(json_source,Source)`

---

generate/2

Generates the representation specified in the first argument (`file(Path)`, `stream(Stream)`, `codes(List)`, `chars(List)`, or `atom(Atom)`) for an OpenAPI document term using JSON serialization.

Compilation flags:
`static`

Template:
`generate(Sink,Document)`
Mode and number of proofs:
`generate(+compound,++term)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid JSON sink term:
`domain_error(json_sink,Sink)`
A JSON version of `Document` cannot be generated by the implementation:
`domain_error(json_sink,Document)`

---

validate_document/1

Succeeds if the given OpenAPI document term is valid according to the phase-1 OpenAPI validation schema.

Compilation flags:
`static`

Template:
`validate_document(Document)`
Mode and number of proofs:
`validate_document(+term)` - `zero_or_one`

---

validate_document/2

Validates an OpenAPI document term and returns a list of validation errors (empty list if valid).

Compilation flags:
`static`

Template:
`validate_document(Document,Errors)`
Mode and number of proofs:
`validate_document(+term,--list)` - `one`

---

operation/3

Resolves an operation descriptor exposed by a provider object using its `operationId` atom.

Compilation flags:
`static`

Template:
`operation(Provider,OperationId,Operation)`
Mode and number of proofs:
`operation(+object_identifier,+atom,-compound)` - `one_or_error`

Exceptions:
`Provider` does not expose the requested `operationId`:
`existence_error(open_api_operation,OperationId)`
`Provider` exposes duplicated `operationId` values:
`domain_error(open_api_operation_id,duplicate(Id))`

---

validate_request/3

Succeeds when the given normalized HTTP request term or object implementing the `http_request_protocol` protocol conforms to the provider operation descriptor identified by the given `operationId`.

Compilation flags:
`static`

Template:
`validate_request(Provider,OperationId,Request)`
Mode and number of proofs:
`validate_request(+object_identifier,+atom,++term)` - `zero_or_one`

---

validate_request/4

Validates a normalized HTTP request term or an object implementing the `http_request_protocol` protocol against the provider operation descriptor identified by the given `operationId` and returns any contract errors.

Compilation flags:
`static`

Template:
`validate_request(Provider,OperationId,Request,Errors)`
Mode and number of proofs:
`validate_request(+object_identifier,+atom,++term,--list)` - `one`

---

validate_http_request/3

Succeeds when an HTTP request wire source parsed by the `http_core` library conforms to the provider operation descriptor identified by the given `operationId`.

Compilation flags:
`static`

Template:
`validate_http_request(Provider,OperationId,Source)`
Mode and number of proofs:
`validate_http_request(+object_identifier,+atom,++compound)` - `zero_or_one`

---

validate_http_request/4

Parses an HTTP request wire source using the `http_core` library and validates the resulting normalized request against the provider operation descriptor identified by the given `operationId`.

Compilation flags:
`static`

Template:
`validate_http_request(Provider,OperationId,Source,Errors)`
Mode and number of proofs:
`validate_http_request(+object_identifier,+atom,++compound,--list)` - `one`

---

validate_response/3

Succeeds when the given normalized HTTP response term or object implementing the `http_response_protocol` protocol conforms to the provider operation descriptor identified by the given `operationId`.

Compilation flags:
`static`

Template:
`validate_response(Provider,OperationId,Response)`
Mode and number of proofs:
`validate_response(+object_identifier,+atom,++term)` - `zero_or_one`

---

validate_response/4

Validates a normalized HTTP response term or an object implementing the `http_response_protocol` protocol against the provider operation descriptor identified by the given `operationId` and returns any contract errors.

Compilation flags:
`static`

Template:
`validate_response(Provider,OperationId,Response,Errors)`
Mode and number of proofs:
`validate_response(+object_identifier,+atom,++term,--list)` - `one`

---

validate_http_response/3

Succeeds when an HTTP response wire source parsed by the `http_core` library conforms to the provider operation descriptor identified by the given `operationId`.

Compilation flags:
`static`

Template:
`validate_http_response(Provider,OperationId,Source)`
Mode and number of proofs:
`validate_http_response(+object_identifier,+atom,++compound)` - `zero_or_one`

---

validate_http_response/4

Parses an HTTP response wire source using the `http_core` library and validates the resulting normalized response against the provider operation descriptor identified by the given `operationId`.

Compilation flags:
`static`

Template:
`validate_http_response(Provider,OperationId,Source,Errors)`
Mode and number of proofs:
`validate_http_response(+object_identifier,+atom,++compound,--list)` - `one`

---

json_media_descriptor/2

Constructs an OpenAPI JSON media descriptor using the default `application/json` media type.

Compilation flags:
`static`

Template:
`json_media_descriptor(Schema,MediaDescriptor)`
Mode and number of proofs:
`json_media_descriptor(++term,-compound)` - `one_or_error`

Exceptions:
`Schema` is a variable:
`instantiation_error`
`Schema` is neither a boolean schema nor a JSON object schema term nor a `schema_ref/1` term:
`domain_error(open_api_json_schema_term,Schema)`

---

json_media_descriptor/3

Constructs an OpenAPI JSON media descriptor using the given media type.

Compilation flags:
`static`

Template:
`json_media_descriptor(MediaType,Schema,MediaDescriptor)`
Mode and number of proofs:
`json_media_descriptor(+atom,++term,-compound)` - `one_or_error`

Exceptions:
`MediaType` is a variable:
`instantiation_error`
`MediaType` is neither a variable nor a JSON-compatible HTTP media type atom:
`domain_error(open_api_media_type,MediaType)`
`Schema` is a variable:
`instantiation_error`
`Schema` is neither a boolean schema nor a JSON object schema term nor a `schema_ref/1` term:
`domain_error(open_api_json_schema_term,Schema)`

---

json_request_body_descriptor/4

Constructs an OpenAPI request-body descriptor using the default `application/json` media type.

Compilation flags:
`static`

Template:
`json_request_body_descriptor(Description,Required,Schema,RequestBody)`
Mode and number of proofs:
`json_request_body_descriptor(+atom,+boolean,++term,-compound)` - `one_or_error`

Exceptions:
`Description` is a variable:
`instantiation_error`
`Description` is neither a variable nor an atom:
`domain_error(open_api_description,Description)`
`Required` is a variable:
`instantiation_error`
`Required` is neither a variable nor a boolean:
`domain_error(open_api_boolean,Required)`
`Schema` is a variable:
`instantiation_error`
`Schema` is neither a boolean schema nor a JSON object schema term nor a `schema_ref/1` term:
`domain_error(open_api_json_schema_term,Schema)`

json_request_body_descriptor/5

Constructs an OpenAPI request-body descriptor using the given media type.

Compilation flags:
`static`

Template:
`json_request_body_descriptor(MediaType,Description,Required,Schema,RequestBody)`
Mode and number of proofs:
`json_request_body_descriptor(+atom,+atom,+boolean,++term,-compound)` - `one_or_error`

Exceptions:
`MediaType` is a variable:
`instantiation_error`
`MediaType` is neither a variable nor a JSON-compatible HTTP media type atom:
`domain_error(open_api_media_type,MediaType)`
`Description` is a variable:
`instantiation_error`
`Description` is neither a variable nor an atom:
`domain_error(open_api_description,Description)`
`Required` is a variable:
`instantiation_error`
`Required` is neither a variable nor a boolean:
`domain_error(open_api_boolean,Required)`
`Schema` is a variable:
`instantiation_error`
`Schema` is neither a boolean schema nor a JSON object schema term nor a `schema_ref/1` term:
`domain_error(open_api_json_schema_term,Schema)`

---

json_response_descriptor/4

Constructs an OpenAPI response descriptor using the default `application/json` media type.

Compilation flags:
`static`

Template:
`json_response_descriptor(Status,Description,Schema,ResponseDescriptor)`
Mode and number of proofs:
`json_response_descriptor(++term,+atom,++term,-compound)` - `one_or_error`

Exceptions:
`Status` is a variable:
`instantiation_error`

`Status` is neither a variable nor `default` nor an integer in the `100..599` range nor a wildcard status atom such as `'2XX'`:
`domain_error(open_api_response_status,Status)`
`Description` is a variable:
`instantiation_error`
`Description` is neither a variable nor an atom:
`domain_error(open_api_description,Description)`
`Schema` is a variable:
`instantiation_error`
`Schema` is neither a boolean schema nor a JSON object schema term nor a `schema_ref/1` term:
`domain_error(open_api_json_schema_term,Schema)`

`json_response_descriptor/5`

Constructs an OpenAPI response descriptor using the given media type.

Compilation flags:
`static`

Template:
`json_response_descriptor(Status,Description,MediaType,Schema,ResponseDescriptor)`
Mode and number of proofs:
`json_response_descriptor(++term,+atom,+atom,++term,-compound)` - `one_or_error`

Exceptions:
`Status` is a variable:
`instantiation_error`
`Status` is neither a variable nor `default` nor an integer in the `100..599` range nor a wildcard status atom such as `'2XX'`:
`domain_error(open_api_response_status,Status)`
`Description` is a variable:
`instantiation_error`
`Description` is neither a variable nor an atom:
`domain_error(open_api_description,Description)`
`MediaType` is a variable:
`instantiation_error`
`MediaType` is neither a variable nor a JSON-compatible HTTP media type atom:
`domain_error(open_api_media_type,MediaType)`
`Schema` is a variable:
`instantiation_error`
`Schema` is neither a boolean schema nor a JSON object schema term nor a `schema_ref/1` term:
`domain_error(open_api_json_schema_term,Schema)`

---

problem_json_schema/1

Returns an inline RFC 7807-compatible JSON schema for problem responses.

Compilation flags:
`static`

Template:
`problem_json_schema(Schema)`
Mode and number of proofs:
`problem_json_schema(-term)` - `one`

---

problem_response_descriptor/3

Constructs an OpenAPI problem-response descriptor using the `application/problem+json` media type and the default problem schema.

Compilation flags:
`static`

Template:
`problem_response_descriptor(Status,Description,ResponseDescriptor)`
Mode and number of proofs:
`problem_response_descriptor(++term,+atom,-compound)` - `one_or_error`

Exceptions:
`Status` is a variable:
`instantiation_error`
`Status` is neither a variable nor `default` nor an integer in the `100..599` range nor a wildcard status atom such as `'2XX'`:
`domain_error(open_api_response_status,Status)`
`Description` is a variable:
`instantiation_error`
`Description` is neither a variable nor an atom:
`domain_error(open_api_description,Description)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

json, *json_schema*, *application_protocol*, *open_api_provider_protocol*

protocol

1.242.2 `open_api_provider_protocol`

Protocol for Logtalk objects that provide metadata used to derive OpenAPI 3.1.0 documents.

Availability:
`logtalk_load(open_api(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *api_info/1*
  - *servers/1*
  - *security/1*
  - *operations/1*
  - *schema/2*
  - *security_scheme/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

api_info/1

Returns the API metadata descriptor. The descriptor must be the compound term `info(Title, Version, Summary, Properties)` where `Title`, `Version`, and `Summary` are atoms and `Properties` is a list of optional metadata terms. Currently recognized properties: `description(Description)` where `Description` is an atom.

Compilation flags:
`static`

Template:
`api_info(Info)`
Mode and number of proofs:
`api_info(-compound)` - `one`

---

servers/1

Returns a list of server descriptors available for this API. Each descriptor must be a compound term `server(URL, Description)` where `URL` is a valid OpenAPI Server Object URL atom, allowing absolute URLs, relative references, and templated variables, and `Description` is a human-readable atom.

Compilation flags:
`static`

Template:
`servers(Servers)`
Mode and number of proofs:
`servers(-list(compound))` - `one`

---

security/1

Returns the top-level OpenAPI security requirement alternatives for this API. The returned list uses the same representation as operation `security(Requirements)` properties: each element is a requirement alternative represented as a list of `Scheme-Scopes` pairs and `[]` denotes an empty security requirement object. Scheme names must be declared by `security_scheme/2`.

Compilation flags:
`static`

Template:
`security(Security)`
Mode and number of proofs:
`security(-list(compound))` - `zero_or_one`

---

operations/1

Returns a list of operation descriptors exposed by this API. Each descriptor must be a compound term `operation(Id, Method, Path, Summary, Parameters, RequestBody, Responses, Properties)` where `Id` is an atom and unique across the returned descriptors, `Method` is a lowercase HTTP method atom, `Path` is a path-template atom, `Summary` is an atom, `Parameters` is a list of `parameter(Name, In, Description, Required, Schema)` terms, `RequestBody` is either `none` or `request_body(Description, Required, MediaTypes)`, `Responses` is a list of `response(Status, Description, MediaTypes)` terms, and `Properties` is a list of optional operation property terms. `Status` may be an integer HTTP status code, `default`, or a wildcard range atom such as `'2XX'`. Path parameters must use `In = path`, match a template expression in `Path`, and use `Required = true`. Currently recognized property terms are `description(Description)`, `tags(Tags)`, `deprecated(Boolean)`, and `security(Requirements)` where `Requirements` is a list whose elements are requirement alternatives represented as lists of `Scheme-Scopes` pairs; the empty list element `[]` denotes an empty security requirement object, scheme names must be declared by `security_scheme/2`, and referenced OAuth flow scopes must be declared by the corresponding security scheme when locally available.

Compilation flags:

`static`

Template:

`operations(Operations)`

Mode and number of proofs:

`operations(-list(compound))` - `one`

---

schema/2

Maps reusable schema names to JSON Schema terms. `Name` must be an atom and `Schema` must be a JSON Schema curly term compatible with the `json_schema` library. This predicate is intended both for lookup by name and for enumeration of all reusable schemas referenced from `schema_ref(Name)` terms.

Compilation flags:

`static`

Template:

`schema(Name,Schema)`

Mode and number of proofs:

`schema(?atom,?compound)` - `zero_or_more`

---

security_scheme/2

Maps reusable security scheme names to security descriptor terms. `Name` must be an atom matching scheme names referenced from operation `security(Requirements)` properties and `Scheme` must use one of the supported descriptor shapes: `api_key/2-3`, `http/1-2`, `mutual_tls/0-1`, `oauth2/1-2`, or `openid_connect/1-2`. OAuth flow descriptors are expressed using `implicit/2-3`, `password/2-3`, `client_credentials/2-3`, and `authorization_code/3-4` with `scope/2` terms. `openid_connect/2` also accepts a provider-only `flows(Flows)` option for local scope declarations used during security-reference validation; those local flow declarations are not emitted into the derived OpenAPI document. Provider-side validation checks required fields, valid `apiKey` locations, and URL-valued fields in OAuth or OpenID Connect descriptors.

Compilation flags:

```
static
```

Template:

```
security_scheme(Name,Scheme)
```

Mode and number of proofs:

```
security_scheme(?atom,?compound) - zero_or_more
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.243 open_id

object

1.243.1 `open_id`

Facade predicates for a portable OpenID Connect Authorization Code + PKCE client.

Availability:

```
logtalk_load(open_id(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-12

Compilation flags:

```
static, context_switching_calls
```

Uses:

open_id_client
open_id_jwks_cache
open_id_jwt
open_id_logout
open_id_pkce
open_id_response

Remarks:

(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *discover/3*
  - *authorization_url/5*
  - *authorization_response/3*
  - *authorization_code/4*
  - *exchange_code/5*
  - *refresh_token/4*
  - *userinfo/4*
  - *logout_url/4*
  - *jwks/3*
  - *cached_jwks/3*
  - *verify_id_token/4*
  - *verify_id_token/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

discover/3

Discovers and validates OpenID Provider metadata for the given issuer.

Compilation flags:
`static`

Template:
`discover(Issuer,Provider,Options)`
Mode and number of proofs:
`discover(+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Issuer` is a variable:
`instantiation_error`
`Issuer` is not an absolute or allowed issuer URL:
`domain_error(open_id_issuer,Issuer)`

The discovery endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The discovery endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`
The discovered issuer does not match `Issuer`:
`domain_error(open_id_issuer,DiscoveredIssuer)`
A provider metadata URL is not secure:
`domain_error(open_id_provider_metadata_url,URL)`

---

authorization_url/5

Builds an authorization URL and returns the session data needed for the code exchange and ID-token validation.

Compilation flags:
`static`

Template:
`authorization_url(Provider,Request,URL,Session,Options)`
Mode and number of proofs:
`authorization_url(+compound,+compound,-atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Request` is neither a direct options list nor a request wrapper term containing an options list:
`domain_error(open_id_options,Request)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Request` is missing a required authorization request option:
`domain_error(open_id_authorization_request,missing(Name))`
A request value that must be an atom is not an atom:
`type_error(atom,Value)`
A scope value is neither an atom nor a list of atoms:
`domain_error(open_id_space_separated_atom,Value)`
The PKCE code verifier is a variable:
`instantiation_error`
The PKCE code verifier is not valid:
`domain_error(open_id_code_verifier,Verifier)`

---

authorization_response/3

Parses an authorization callback URL into either an authorization response or an authorization error term.

Compilation flags:

```
static
```

Template:

```
authorization_response(CallbackURL,Response,Options)
```

Mode and number of proofs:

```
authorization_response(+atom,-compound,+list(compound)) - one_or_error
```

Exceptions:

`Options` is a variable or a partial list:

```
instantiation_error
```

`Options` is neither a variable nor a list:

```
type_error(list,Options)
```

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

`CallbackURL` is not a valid callback URL or relative reference:

```
domain_error(open_id_authorization_response,CallbackURL)
```

---

authorization_code/4

Parses an authorization callback URL, validates the session state, and returns the authorization code.

Compilation flags:

```
static
```

Template:

```
authorization_code(CallbackURL,Session,Code,Options)
```

Mode and number of proofs:

```
authorization_code(+atom,+compound,-atom,+list(compound)) - one_or_error
```

Exceptions:

`Options` is a variable or a partial list:

```
instantiation_error
```

`Options` is neither a variable nor a list:

```
type_error(list,Options)
```

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

`Session` is missing required state data:

```
domain_error(open_id_session,missing(Session,state))
```

The callback response is an authorization error:
`domain_error(open_id_authorization_response,authorization_error(Response))`
The callback response state does not match the session state:
`domain_error(open_id_authorization_response,state_mismatch(Expected,Actual))`

---

exchange_code/5

Exchanges an authorization code for token response data.

Compilation flags:
`static`

Template:
`exchange_code(Provider,Code,Session,Tokens,Options)`
Mode and number of proofs:
`exchange_code(+compound,+atom,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Session` is missing required authorization-code exchange data:
`domain_error(open_id_session,missing(Session,Name))`
The token endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The token endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`
The token endpoint JSON response is missing a required member:
`domain_error(open_id_token_response,missing(Name))`

---

refresh_token/4

Exchanges a refresh token for token response data.

Compilation flags:
`static`

Template:
`refresh_token(Provider,RefreshToken,Tokens,Options)`
Mode and number of proofs:

`refresh_token(+compound,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The `client_id` option is missing:
`domain_error(open_id_refresh_token,missing(client_id))`
The token endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The token endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`
The token endpoint JSON response is missing a required member:
`domain_error(open_id_token_response,missing(Name))`

`userinfo/4`

Fetches UserInfo claims using the given access token.

Compilation flags:

`static`

Template:

`userinfo(Provider,AccessToken,Claims,Options)`

Mode and number of proofs:

`userinfo(+compound,+atom,-term,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The provider metadata is missing a UserInfo endpoint:
`domain_error(open_id_provider,missing(userinfo_endpoint))`
The UserInfo endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The UserInfo endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`

logout_url/4

Builds a logout URL for an RP-initiated logout request.

Compilation flags:
`static`

Template:
`logout_url(Provider,Request,URL,Options)`
Mode and number of proofs:
`logout_url(+compound,+compound,-atom,+list(compound))` - `one_or_error`

Exceptions:
`Request` is neither a direct options list nor a request wrapper term containing an options list:
`domain_error(open_id_options,Request)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The provider metadata is missing an end-session endpoint:
`domain_error(open_id_provider,missing(end_session_endpoint))`
A request value that must be an atom is not an atom:
`type_error(atom,Value)`
A `ui_locales` value is neither an atom nor a list of atoms:
`domain_error(open_id_space_separated_atom,Value)`
The `post_logout_redirect_uri` is not secure:
`domain_error(open_id_post_logout_redirect_uri,URL)`

jwks/3

Fetches the JSON Web Key Set for the provider.

Compilation flags:
`static`

Template:
`jwks(Provider,JWKSet,Options)`
Mode and number of proofs:
`jwks(+compound,-term,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The JWKS endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The JWKS endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`

`cached_jwks/3`

Returns a cached JWKS for a provider, refreshing it when needed.

Compilation flags:
`static`

Template:
`cached_jwks(Provider,JWKSet,Options)`
Mode and number of proofs:
`cached_jwks(+compound,-term,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The provider metadata is missing a JWKS endpoint:
`domain_error(open_id_provider,missing(jwks_uri))`
The JWKS endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The JWKS endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`

---

verify_id_token/4

Verifies an ID-token using cached JWKS data, refreshing the JWKS once when the token key identifier is unknown.

Compilation flags:

`static`

Template:

`verify_id_token(Token,Provider,Claims,Options)`

Mode and number of proofs:

`verify_id_token(+atom,+compound,-term,+list(compound))` - `zero_or_one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The provider metadata is missing a JWKS endpoint:

`domain_error(open_id_provider,missing(jwks_uri))`

The JWKS endpoint returned a non-success status:

`domain_error(open_id_http_status,Code)`

The JWKS endpoint response body is not JSON:

`domain_error(open_id_json_response,Body)`

`Token` is a variable:

`instantiation_error`

`Token` is neither a variable nor an atom:

`type_error(atom,Token)`

`Token` is an atom but not a valid compact JWT:

`domain_error(jwt_compact_serialization,Token)`

`Token` contains Base64URL data with characters outside the Base64URL alphabet:

`representation_error(base64)`

The JWT header does not contain a valid `alg` member:

`domain_error(jwt_header,Header)`

The JWT algorithm is unsupported or disallowed:

`domain_error(open_id_jwt_algorithm,Algorithm)`

`JWKSet` is not a JSON Web Key Set with a `keys` list:

`domain_error(jwt_jwks,JWKSet)`

No compatible key exists for the JWT header:

`existence_error(jwt_jwk,Header)`

The selected JWK is not a supported public key:

`domain_error(jwt_jwk_public_key,Key)`

The ES256 signature is not 64 bytes long:

`domain_error(jwt_es256_signature,Signature)`

The OpenSSL executable does not exist:

`existence_error(os_command,Executable)`

The `iss` claim does not match the provider issuer:

`domain_error(open_id_claim(iss),Value)`

A required OpenID ID-token claim is missing:

`domain_error(open_id_claims,missing(Name))`

The expected audience option is missing:

`domain_error(open_id_claims,missing(expected_audience))`

An OpenID ID-token claim has an invalid value:

`domain_error(open_id_claim(Name),Value)`

An OpenID ID-token time claim value is not a number:

`type_error(time_number,Name-Time)`

---

verify_id_token/5

Verifies an ID-token signature and claims using the given provider metadata and JWKS.

Compilation flags:

`static`

Template:

`verify_id_token(Token,Provider,JWKSet,Claims,Options)`

Mode and number of proofs:

`verify_id_token(+atom,+compound,+term,-term,+list(compound))` - `zero_or_one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Token` is a variable:

`instantiation_error`

`Token` is neither a variable nor an atom:

`type_error(atom,Token)`

`Token` is an atom but not a valid compact JWT:

`domain_error(jwt_compact_serialization,Token)`

`Token` contains Base64URL data with characters outside the Base64URL alphabet:

`representation_error(base64)`

The JWT header does not contain a valid `alg` member:

`domain_error(jwt_header,Header)`

The JWT algorithm is unsupported or disallowed:

`domain_error(open_id_jwt_algorithm,Algorithm)`

`JWKSet` is not a JSON Web Key Set with a `keys` list:

`domain_error(jwt_jwks,JWKSet)`

No compatible key exists for the JWT header:

`existence_error(jwt_jwk,Header)`
The selected JWK is not a supported public key:
`domain_error(jwt_jwk_public_key,Key)`
The ES256 signature is not 64 bytes long:
`domain_error(jwt_es256_signature,Signature)`
The OpenSSL executable does not exist:
`existence_error(os_command,Executable)`
The `iss` claim does not match the provider issuer:
`domain_error(open_id_claim(iss),Value)`
A required OpenID ID-token claim is missing:
`domain_error(open_id_claims,missing(Name))`
The expected audience option is missing:
`domain_error(open_id_claims,missing(expected_audience))`
An OpenID ID-token claim has an invalid value:
`domain_error(open_id_claim(Name),Value)`
An OpenID ID-token time claim value is not a number:
`type_error(time_number,Name-Time)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.243.2 `open_id_client`

HTTP-facing OpenID Connect client predicates.

Availability:
`logtalk_load(open_id(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *open_id_helpers*
Uses:

base64
http_client
http_core
list
open_id_discovery
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *discover/3*
  - *jwks/3*
  - *exchange_code/5*
  - *refresh_token/4*
  - *userinfo/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

discover/3

Discovers and validates OpenID Provider metadata for the given issuer.

Compilation flags:
`static`

Template:
`discover(Issuer,Provider,Options)`
Mode and number of proofs:
`discover(+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Issuer` is a variable:
`instantiation_error`

`Issuer` is not an absolute or allowed issuer URL:
`domain_error(open_id_issuer,Issuer)`
The discovery endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The discovery endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`
The discovered issuer does not match `Issuer`:
`domain_error(open_id_issuer,DiscoveredIssuer)`
A provider metadata URL is not secure:
`domain_error(open_id_provider_metadata_url,URL)`

`jwks/3`

Fetches the JSON Web Key Set for the given provider metadata.

Compilation flags:
`static`

Template:
`jwks(Provider,JWKSet,Options)`
Mode and number of proofs:
`jwks(+compound,-term,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The JWKS endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The JWKS endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`

---

exchange_code/5

Exchanges an authorization code for token response data.

Compilation flags:
`static`

Template:
`exchange_code(Provider,Code,Session,Tokens,Options)`
Mode and number of proofs:
`exchange_code(+compound,+atom,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Session` is missing required authorization-code exchange data:
`domain_error(open_id_session,missing(Session,Name))`
The token endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The token endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`
The token endpoint JSON response is missing a required member:
`domain_error(open_id_token_response,missing(Name))`

---

refresh_token/4

Exchanges a refresh token for token response data.

Compilation flags:
`static`

Template:
`refresh_token(Provider,RefreshToken,Tokens,Options)`
Mode and number of proofs:
`refresh_token(+compound,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The `client_id` option is missing:
`domain_error(open_id_refresh_token,missing(client_id))`
The token endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The token endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`
The token endpoint JSON response is missing a required member:
`domain_error(open_id_token_response,missing(Name))`

`userinfo/4`

Fetches UserInfo claims using the given access token.

Compilation flags:
`static`

Template:
`userinfo(Provider,AccessToken,Claims,Options)`
Mode and number of proofs:
`userinfo(+compound,+atom,-term,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The provider metadata is missing a UserInfo endpoint:
`domain_error(open_id_provider,missing(userinfo_endpoint))`
The UserInfo endpoint returned a non-success status:
`domain_error(open_id_http_status,Code)`
The UserInfo endpoint response body is not JSON:
`domain_error(open_id_json_response,Body)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.243.3 `open_id_der`

DER and PEM helpers for OpenID Connect JWT signature verification.

Availability:
`logtalk_load(open_id(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-08

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *open_id_helpers*
Uses:
base64
list
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *public_key_pem/2*
  - *es256_signature_der/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

public_key_pem/2

Converts a supported JWK public key into PEM SubjectPublicKeyInfo text.

Compilation flags:
`static`

Template:
`public_key_pem(Key,PEM)`
Mode and number of proofs:
`public_key_pem(+term,-atom)` - `one_or_error`

Exceptions:
`Key` is not a supported RSA or P-256 EC public JWK:
`domain_error(open_id_jwk_public_key,Key)`
`Key` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

---

es256_signature_der/2

Converts a raw ES256 signature into DER-encoded ASN.1 form.

Compilation flags:
`static`

Template:
`es256_signature_der(Signature,DER)`
Mode and number of proofs:
`es256_signature_der(+list(byte),-list(byte))` - `one_or_error`

Exceptions:
`Signature` is not a 64-byte raw ES256 signature:
`domain_error(open_id_es256_signature,Signature)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.243.4 `open_id_discovery`

OpenID Provider discovery metadata helpers.

Availability:
`logtalk_load(open_id(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-08

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *open_id_helpers*
Uses:
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *discovery_url/2*
  - *provider/3*
  - *provider/4*
  - *property/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

discovery_url/2

Builds the discovery metadata URL for an issuer.

Compilation flags:
`static`

Template:
`discovery_url(Issuer,URL)`
Mode and number of proofs:
`discovery_url(+atom,-atom)` - `one_or_error`

Exceptions:
`Issuer` is a variable:
`instantiation_error`
`Issuer` is not an absolute issuer URL:
`domain_error(open_id_issuer,Issuer)`

---

provider/3

Builds validated provider metadata from discovery JSON using default options.

Compilation flags:
`static`

Template:
`provider(ExpectedIssuer,JSON,Provider)`
Mode and number of proofs:
`provider(+atom,+term,-compound)` - `one_or_error`

Exceptions:
The discovered issuer does not match `ExpectedIssuer`:
`domain_error(open_id_issuer,Issuer)`
A provider metadata URL is not secure:
`domain_error(open_id_provider_metadata_url,URL)`

---

provider/4

Builds validated provider metadata from discovery JSON using the given options.

Compilation flags:
`static`

Template:
`provider(ExpectedIssuer,JSON,Provider,Options)`
Mode and number of proofs:
`provider(+atom,+term,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The discovered issuer does not match `ExpectedIssuer`:
`domain_error(open_id_issuer,Issuer)`
A provider metadata URL is not secure:
`domain_error(open_id_provider_metadata_url,URL)`

---

property/3

Looks up a property in validated provider metadata.

Compilation flags:
`static`

Template:
`property(Provider,Name,Value)`
Mode and number of proofs:
`property(+compound,+atom,-term)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.243.5 `open_id_helpers`

Internal shared helpers for the open_id library.

Availability:
`logtalk_load(open_id(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:
`static`

Extends:
`public` *options*
Uses:
base64url_no_padding
http_core
list
url(Representation)
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *json_member/3*
  - *json_member_default/4*
  - *provider_property/3*
  - *provider_property_default/4*
  - *normalize_options_term/2*

- query_atom/2
- space_atom/2
- url_scheme/2
- https_url/1
- localhost_http_url/1
- ensure_absolute_url/2
- ensure_secure_url/3
- atom_bytes/2
- bytes_atom/2
- base64url_atom_bytes/2
- json_object_pairs/2
- pair_key_value/3
- append_query/3

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

json_member/3

Enumerates JSON object member values for a given key.

Compilation flags:
`static`

Template:
`json_member(Key,Object,Value)`

Mode and number of proofs:
`json_member(+atom,+term,-term)` - `zero_or_more`

---

json_member_default/4

Returns a JSON object member value or a default when the key is absent.

Compilation flags:
`static`

Template:
`json_member_default(Key,Object,Default,Value)`

Mode and number of proofs:
`json_member_default(+atom,+term,+term,-term)` - `one`

---

provider_property/3

Looks up a property in a provider metadata compound term.

Compilation flags:
```
static
```

Template:
```
provider_property(Provider,Name,Value)
```
Mode and number of proofs:
```
provider_property(+compound,+atom,-term) - zero_or_one
```

---

provider_property_default/4

Returns a provider metadata property or a default when the property is absent.

Compilation flags:
```
static
```

Template:
```
provider_property_default(Provider,Name,Default,Value)
```
Mode and number of proofs:
```
provider_property_default(+compound,+atom,+term,-term) - one
```

---

normalize_options_term/2

Normalizes either a direct options list or a named options wrapper term.

Compilation flags:
```
static
```

Template:
```
normalize_options_term(NameOptions,Options)
```
Mode and number of proofs:
```
normalize_options_term(+term,-list(compound)) - one
```

---

query_atom/2

Encodes query parameters as a URL query atom.

Compilation flags:
`static`

Template:
`query_atom(Pairs,Query)`
Mode and number of proofs:
`query_atom(+list(pair),-atom)` - `one_or_error`

Exceptions:
The HTTP body sink is a variable:
`instantiation_error`
The HTTP body sink is not a valid sink term:
`domain_error(http_sink,Sink)`
`Pairs` cannot be represented as an HTTP form body:
`domain_error(http_body,Body)`
The HTTP body options are invalid:
`domain_error(http_body_options,Options)`

---

space_atom/2

Converts an atom or list of atoms into a space-separated atom.

Compilation flags:
`static`

Template:
`space_atom(Value,Atom)`
Mode and number of proofs:
`space_atom(+term,-atom)` - `one_or_error`

Exceptions:
`Value` is neither an atom nor a list of atoms:
`domain_error(open_id_space_separated_atom,Value)`

---

url_scheme/2

Extracts the scheme from an absolute URL atom.

Compilation flags:
`static`

Template:
`url_scheme(URL,Scheme)`
Mode and number of proofs:
`url_scheme(+atom,-atom)` - `zero_or_one`

---

https_url/1

Succeeds when the URL uses the https scheme.

Compilation flags:
`static`

Template:
`https_url(URL)`
Mode and number of proofs:
`https_url(+atom)` - `zero_or_one`

---

localhost_http_url/1

Succeeds when the URL uses http for a localhost or loopback host.

Compilation flags:
`static`

Template:
`localhost_http_url(URL)`
Mode and number of proofs:
`localhost_http_url(+atom)` - `zero_or_one`

---

ensure_absolute_url/2

Validates that a value is an absolute URL for the given domain label.

Compilation flags:

```
static
```

Template:

```
ensure_absolute_url(Domain,URL)
```

Mode and number of proofs:

```
ensure_absolute_url(+atom,+atom) - one_or_error
```

Exceptions:

`URL` is a variable:

```
instantiation_error
```

`URL` is not an absolute URL atom:

```
domain_error(Domain,URL)
```

---

ensure_secure_url/3

Validates that a URL uses https or an allowed localhost exception.

Compilation flags:

```
static
```

Template:

```
ensure_secure_url(Domain,URL,Options)
```

Mode and number of proofs:

```
ensure_secure_url(+atom,+atom,+list(compound)) - one_or_error
```

Exceptions:

`URL` is a variable:

```
instantiation_error
```

`URL` is not an absolute URL atom:

```
domain_error(Domain,URL)
```

`URL` is not secure and is not an allowed localhost HTTP URL:

```
domain_error(Domain,URL)
```

---

atom_bytes/2

Converts between an atom and its byte list representation.

Compilation flags:
`static`

Template:
`atom_bytes(Atom,Bytes)`
Mode and number of proofs:
`atom_bytes(+atom,-list(byte))` - `one`
`atom_bytes(-atom,+list(byte))` - `one`

---

bytes_atom/2

Converts a byte list into an atom.

Compilation flags:
`static`

Template:
`bytes_atom(Bytes,Atom)`
Mode and number of proofs:
`bytes_atom(+list(byte),-atom)` - `one`

---

base64url_atom_bytes/2

Converts between unpadded Base64URL atoms and byte lists.

Compilation flags:
`static`

Template:
`base64url_atom_bytes(Atom,Bytes)`
Mode and number of proofs:
`base64url_atom_bytes(+atom,-list(byte))` - `one_or_error`
`base64url_atom_bytes(-atom,+list(byte))` - `one_or_error`

Exceptions:
`Atom` is a variable or the sink is a variable:
`instantiation_error`
`Atom` is neither a variable nor a valid Base64URL source term:
`domain_error(base64url_source,codes(Codes))`
`Bytes` cannot be encoded in the requested Base64URL sink representation:
`domain_error(base64url_sink,codes(Codes))`

`Atom` contains Base64URL data with characters outside the Base64URL alphabet:
`representation_error(base64)`

`json_object_pairs/2`

Extracts the key-value pairs from a JSON object term.

Compilation flags:
`static`

Template:
`json_object_pairs(Object,Pairs)`
Mode and number of proofs:
`json_object_pairs(+term,-list(compound))` - `zero_or_one`

`pair_key_value/3`

Extracts the key and value from a JSON pair compound term.

Compilation flags:
`static`

Template:
`pair_key_value(Pair,Key,Value)`
Mode and number of proofs:
`pair_key_value(+compound,-term,-term)` - `one`

`append_query/3`

Appends a query atom to an endpoint URL, preserving any existing query.

Compilation flags:
`static`

Template:
`append_query(Endpoint,Query,URL)`
Mode and number of proofs:
`append_query(+atom,+atom,-atom)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.243.6 `open_id_jwks`

JSON Web Key Set parsing and key selection helpers.

Availability:
`logtalk_load(open_id(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-08

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *open_id_helpers*
Uses:
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *select_key/3*
  - *header_algorithm/2*
  - *header_key_id/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

select_key/3

Selects a matching verification key from a JWK Set for a JWT header.

Compilation flags:
`static`

Template:
`select_key(JWKSet,Header,Key)`
Mode and number of proofs:
`select_key(+term,+term,-term)` - `one_or_error`

Exceptions:
`JWKSet` is not a JSON Web Key Set with a `keys` list:
`domain_error(open_id_jwks,JWKSet)`
`Header` does not contain a valid `alg` member:
`domain_error(open_id_jwt_header,Header)`
No compatible key exists for `Header`:
`existence_error(open_id_jwk,Header)`

---

header_algorithm/2

Extracts and validates the JWT header algorithm.

Compilation flags:
`static`

Template:
`header_algorithm(Header,Algorithm)`
Mode and number of proofs:
`header_algorithm(+term,-atom)` - `one_or_error`

Exceptions:
`Header` does not contain a valid `alg` member:
`domain_error(open_id_jwt_header,Header)`

---

header_key_id/2

Extracts the optional key identifier from a JWT header.

Compilation flags:
`static`

Template:
`header_key_id(Header,KeyId)`
Mode and number of proofs:
`header_key_id(+term,-atom)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.243.7 open_id_jwks_cache

JWKS caching and key-rotation helpers.

Availability:
`logtalk_load(open_id(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-12

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *open_id_helpers*
Uses:
open_id_client
open_id_jwt
os

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *cached_jwks/3*
  - *verify_id_token/4*
  - *clear/0*
  - *clear/1*
- *Protected predicates*
- *Private predicates*
  - *cached_jwks_/3*
- *Operators*

Public predicates

cached_jwks/3

Returns a cached JWKS for a provider, refreshing it when the cache is stale or missing.

Compilation flags:

`static`

Template:

`cached_jwks(Provider,JWKSet,Options)`

Mode and number of proofs:

`cached_jwks(+compound,-term,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The provider metadata is missing a JWKS endpoint:

`domain_error(open_id_provider,missing(jwks_uri))`

The JWKS endpoint returned a non-success status:

`domain_error(open_id_http_status,Code)`

The JWKS endpoint response body is not JSON:

`domain_error(open_id_json_response,Body)`

verify_id_token/4

Verifies an ID-token using cached JWKS data, refreshing the cache once when the token key identifier is unknown.

Compilation flags:

`static`

Template:

`verify_id_token(Token,Provider,Claims,Options)`

Mode and number of proofs:

`verify_id_token(+atom,+compound,-term,+list(compound))` - `zero_or_one_or_error`

Exceptions:

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The provider metadata is missing a JWKS endpoint:

`domain_error(open_id_provider,missing(jwks_uri))`

The JWKS endpoint returned a non-success status:

`domain_error(open_id_http_status,Code)`

The JWKS endpoint response body is not JSON:

`domain_error(open_id_json_response,Body)`

`Token` is a variable:

`instantiation_error`

`Token` is neither a variable nor an atom:

`type_error(atom,Token)`

`Token` is an atom but not a valid compact JWT:

`domain_error(jwt_compact_serialization,Token)`

`Token` contains Base64URL data with characters outside the Base64URL alphabet:

`representation_error(base64)`

The JWT header does not contain a valid `alg` member:

`domain_error(jwt_header,Header)`

The JWT algorithm is unsupported or disallowed:

`domain_error(open_id_jwt_algorithm,Algorithm)`

`JWKSet` is not a JSON Web Key Set with a `keys` list:

`domain_error(jwt_jwks,JWKSet)`

No compatible key exists for the JWT header:

`existence_error(jwt_jwk,Header)`

The selected JWK is not a supported public key:

`domain_error(jwt_jwk_public_key,Key)`

The ES256 signature is not 64 bytes long:

`domain_error(jwt_es256_signature,Signature)`

The OpenSSL executable does not exist:

`existence_error(os_command,Executable)`

The `iss` claim does not match the provider issuer:

```
domain_error(open_id_claim(iss),Value)
```

A required OpenID ID-token claim is missing:

```
domain_error(open_id_claims,missing(Name))
```

The expected audience option is missing:

```
domain_error(open_id_claims,missing(expected_audience))
```

An OpenID ID-token claim has an invalid value:

```
domain_error(open_id_claim(Name),Value)
```

An OpenID ID-token time claim value is not a number:

```
type_error(time_number,Name-Time)
```

---

clear/0

Clears all cached JWKS entries.

Compilation flags:

```
static
```

Mode and number of proofs:

```
clear - one
```

---

clear/1

Clears the cached JWKS entry for a provider.

Compilation flags:

```
static
```

Template:

```
clear(Provider)
```

Mode and number of proofs:

```
clear(+compound) - one_or_error
```

Exceptions:

The provider metadata is missing a JWKS endpoint:

```
domain_error(open_id_provider,missing(jwks_uri))
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

cached_jwks_/3

Cached JWKS entries indexed by URI and fetch time.

Compilation flags:
dynamic

Template:
cached_jwks_(URI,FetchedAt,JWKSet)
Mode and number of proofs:
cached_jwks_(?atom,?number,?term) - zero_or_more

Operators

(none)
object

1.243.8 open_id_jwt

JWT parsing and ID-token verification helpers.

Availability:
logtalk_load(open_id(loader))

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-08

Compilation flags:
static, context_switching_calls

Imports:
public *open_id_helpers*
Uses:
jwt
jwt_jwa
list
os

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *decode/3*
  - *verify_id_token/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

decode/3

Decodes a JWT into its header and claims JSON terms without verifying the signature.

Compilation flags:

```
static
```

Template:

```
decode(Token,Header,Claims)
```

Mode and number of proofs:

```
decode(+atom,-term,-term) - one_or_error
```

Exceptions:

`Token` is a variable:

```
instantiation_error
```

`Token` is neither a variable nor an atom:

```
type_error(atom,Token)
```

`Token` is an atom but not a valid compact JWT:

```
domain_error(jwt_compact_serialization,Token)
```

`Token` contains Base64URL data with characters outside the Base64URL alphabet:

```
representation_error(base64)
```

---

verify_id_token/5

Verifies an ID-token signature and claims using the given provider metadata and JWKS.

Compilation flags:

```
static
```

Template:

```
verify_id_token(Token,Provider,JWKSet,Claims,Options)
```

Mode and number of proofs:

```
verify_id_token(+atom,+compound,+term,-term,+list(compound)) -
zero_or_one_or_error
```

Exceptions:

`Options` is a variable or a partial list:
```
instantiation_error
```
`Options` is neither a variable nor a list:
```
type_error(list,Options)
```
An element `Option` of the list `Options` is neither a variable nor a compound term:
```
type_error(compound,Option)
```
An element `Option` of the list `Options` is a compound term but not a valid option:
```
domain_error(option,Option)
```
`Token` is a variable:
```
instantiation_error
```
`Token` is neither a variable nor an atom:
```
type_error(atom,Token)
```
`Token` is an atom but not a valid compact JWT:
```
domain_error(jwt_compact_serialization,Token)
```
`Token` contains Base64URL data with characters outside the Base64URL alphabet:
```
representation_error(base64)
```
The JWT header does not contain a valid `alg` member:
```
domain_error(jwt_header,Header)
```
The JWT algorithm is unsupported or disallowed:
```
domain_error(open_id_jwt_algorithm,Algorithm)
```
`JWKSet` is not a JSON Web Key Set with a `keys` list:
```
domain_error(jwt_jwks,JWKSet)
```
No compatible key exists for the JWT header:
```
existence_error(jwt_jwk,Header)
```
The selected JWK is not a supported public key:
```
domain_error(jwt_jwk_public_key,Key)
```
The ES256 signature is not 64 bytes long:
```
domain_error(jwt_es256_signature,Signature)
```
The OpenSSL executable does not exist:
```
existence_error(os_command,Executable)
```
The `iss` claim does not match the provider issuer:
```
domain_error(open_id_claim(iss),Value)
```
A required OpenID ID-token claim is missing:
```
domain_error(open_id_claims,missing(Name))
```
The expected audience option is missing:
```
domain_error(open_id_claims,missing(expected_audience))
```
An OpenID ID-token claim has an invalid value:
```
domain_error(open_id_claim(Name),Value)
```
An OpenID ID-token time claim value is not a number:
```
type_error(time_number,Name-Time)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.243.9 `open_id_logout`

RP-initiated logout URL helpers for OpenID Connect clients.

Availability:
`logtalk_load(open_id(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-08

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *open_id_helpers*
Uses:
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *logout_url/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

logout_url/4

Builds a logout URL for an RP-initiated logout request.

Compilation flags:

`static`

Template:

`logout_url(Provider,Request,URL,Options)`

Mode and number of proofs:

`logout_url(+compound,+compound,-atom,+list(compound))` - `one_or_error`

Exceptions:

`Request` is neither a direct options list nor a request wrapper term containing an options list:

`domain_error(open_id_options,Request)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The provider metadata is missing an end-session endpoint:

`domain_error(open_id_provider,missing(end_session_endpoint))`

A request value that must be an atom is not an atom:

`type_error(atom,Value)`

A `ui_locales` value is neither an atom nor a list of atoms:

`domain_error(open_id_space_separated_atom,Value)`

The `post_logout_redirect_uri` is not secure:

`domain_error(open_id_post_logout_redirect_uri,URL)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.243.10 open_id_openssl

OpenSSL-backed JWT signature verification helpers.

Availability:
`logtalk_load(open_id(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-17

Compilation flags:
`static, context_switching_calls`

Uses:
crypto
list
open_id_der
os
process

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *verify/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

verify/5

Verifies a JWT signing input and signature using OpenSSL and a PEM public key.

Compilation flags:
`static`

Template:

```
verify(Algorithm,PEM,SigningInput,Signature,Options)
```

Mode and number of proofs:

```
verify(+atom,+atom,+atom,+list(byte),+list(compound)) - zero_or_one_or_error
```

Exceptions:

The OpenSSL executable does not exist:

```
existence_error(os_command,Executable)
```

`Signature` is not a 64-byte raw ES256 signature:

```
domain_error(open_id_es256_signature,Signature)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.243.11 `open_id_pkce`

PKCE and authorization-request helpers for OpenID Connect clients.

Availability:

```
logtalk_load(open_id(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-07

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public open_id_helpers
```

Uses:

crypto
sha256

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *code_verifier/2*
  - *code_challenge/2*
  - *authorization_url/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

code_verifier/2

Returns a valid PKCE code verifier, either provided in options or freshly generated.

Compilation flags:

```
static
```

Template:

```
code_verifier(Verifier,Options)
```

Mode and number of proofs:

```
code_verifier(-atom,+list(compound)) - one_or_error
```

Exceptions:

`Options` is a variable or a partial list:

```
instantiation_error
```

`Options` is neither a variable nor a list:

```
type_error(list,Options)
```

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

`Verifier` is a variable:

```
instantiation_error
```

`Verifier` is not a valid PKCE code verifier:

```
domain_error(open_id_code_verifier,Verifier)
```

---

code_challenge/2

Computes the S256 PKCE code challenge for a code verifier.

Compilation flags:

```
static
```

Template:

```
code_challenge(Verifier,Challenge)
```

Mode and number of proofs:

`code_challenge(+atom,-atom)` - `one_or_error`

Exceptions:

`Verifier` is a variable:

`instantiation_error`

`Verifier` is not a valid PKCE code verifier:

`domain_error(open_id_code_verifier,Verifier)`

`authorization_url/5`

Builds an authorization URL and session data for an Authorization Code plus PKCE flow.

Compilation flags:

`static`

Template:

`authorization_url(Provider,Request,URL,Session,Options)`

Mode and number of proofs:

`authorization_url(+compound,+compound,-atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Request` is neither a direct options list nor a request wrapper term containing an options list:

`domain_error(open_id_options,Request)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Request` is missing a required authorization request option:

`domain_error(open_id_authorization_request,missing(Name))`

A request value that must be an atom is not an atom:

`type_error(atom,Value)`

A scope value is neither an atom nor a list of atoms:

`domain_error(open_id_space_separated_atom,Value)`

The PKCE code verifier is a variable:

`instantiation_error`

The PKCE code verifier is not valid:

`domain_error(open_id_code_verifier,Verifier)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.243.12 `open_id_response`

Authorization callback parsing and state validation helpers.

Availability:
`logtalk_load(open_id(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-08

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *open_id_helpers*
Uses:
http_core
list
url(Representation)
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *authorization_response/3*
  - *authorization_code/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

authorization_response/3

Parses an authorization callback URL into either an authorization response or an authorization error term.

Compilation flags:
`static`

Template:
`authorization_response(CallbackURL,Response,Options)`
Mode and number of proofs:
`authorization_response(+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`CallbackURL` is a variable:
`instantiation_error`
`CallbackURL` is not a valid URL or relative reference:
`domain_error(open_id_authorization_response,CallbackURL)`
The callback URL does not contain a query component and fragment parsing is not allowed:
`domain_error(open_id_authorization_response,missing(parameters))`
The callback response is missing a required member:
`domain_error(open_id_authorization_response,missing(Name))`
The callback issuer does not match the provider issuer:
`domain_error(open_id_authorization_response,issuer_mismatch(Expected,Actual))`

---

authorization_code/4

Parses an authorization callback URL, validates the returned state against the session, and returns the authorization code.

Compilation flags:
`static`

Template:
`authorization_code(CallbackURL,Session,Code,Options)`
Mode and number of proofs:
`authorization_code(+atom,+compound,-atom,+list(compound))` - `one_or_error`

Exceptions:

`Options` is a variable or a partial list:
```
instantiation_error
```
`Options` is neither a variable nor a list:
```
type_error(list,Options)
```
An element `Option` of the list `Options` is neither a variable nor a compound term:
```
type_error(compound,Option)
```
An element `Option` of the list `Options` is a compound term but not a valid option:
```
domain_error(option,Option)
```
`Session` is missing required state data:
```
domain_error(open_id_session,missing(Session,state))
```
The callback response is an authorization error:
```
domain_error(open_id_authorization_response,authorization_error(Response))
```
The callback response state does not match the session state:
```
domain_error(open_id_authorization_response,state_mismatch(Expected,Actual))
```
The callback response is missing a required member:
```
domain_error(open_id_authorization_response,missing(Name))
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.244 optics_clusterer

object

1.244.1 `optics_clusterer`

OPTICS clusterer for continuous datasets. Learns an ordering from a dataset object implementing the `clustering_dataset_protocol` protocol and extracts clusters using an epsilon threshold so the result can be used with the standard clusterer protocol.

Availability:
```
logtalk_load(optics_clusterer(loader))
```

Author: Paulo Moura
Version: 1:3:0
Date: 2026-11-19

Compilation flags:
```
static, context_switching_calls
```

Imports:

`public` *clusterer_common*
`public` *search_indexing*

Uses:

avltree
binary_heap_min
format
list
numberlist
type

Remarks:

(none)

Inherited public predicates:

check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *dbscan_clusterer*

1.245 optionals

object

1.245.1 `maybe`

Types and predicates for type-checking and handling optional terms. Inspired by Haskell.

Availability:
`logtalk_load(optionals(loader))`

Author: Paulo Moura
Version: 0:9:0
Date: 2025-06-19

Compilation flags:
`static, context_switching_calls`

Provides:
type::type/1
type::check/2
arbitrary::arbitrary/1
arbitrary::arbitrary/2
arbitrary::shrinker/1
arbitrary::shrink/3
arbitrary::edge_case/2

Uses:
optional
optional(Optional)
random
type

Remarks:

- Type-checking support: Defines type `maybe(Type)` for checking optional terms where the value hold by the optional term must be of the given type.
- QuickCheck support: Defines clauses for the `arbitrary::arbitrary/1-2`, `arbitrary::shrinker/1`, `arbitrary::shrink/3`, and `arbitrary::edge_case/2` predicates to allow generating random values for the `maybe(Type)` type.

Inherited public predicates:
(none)

- *Public predicates*
  - *cat/2*
  - *sequence/2*
  - *traverse/3*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

cat/2

Returns the values stored in the non-empty optional terms.

Compilation flags:
`static`

Template:
`cat(Optionals,Values)`
Mode and number of proofs:
`cat(+list(optional),-list)` - `one`

---

sequence/2

Returns an optional term with a list of all values when all optional terms are not empty. Otherwise returns an empty optional term.

Compilation flags:
`static`

Template:
`sequence(Optionals,Optional)`
Mode and number of proofs:
`sequence(+list(optional),--nonvar)` - `one`

---

traverse/3

Applies a closure to each list element to generate optional terms and then sequences them into a single optional term holding all values or an empty optional term.

Compilation flags:
`static`

Template:
`traverse(Closure,Terms,Optional)`
Meta-predicate template:
`traverse(2,*,*)`
Mode and number of proofs:
`traverse(+callable,+list,--nonvar)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

optional, *optional(Optional)*, *type*, *arbitrary*

object

1.245.2 `optional`

Constructors for optional terms. An optional term is either empty or holds a value. Optional terms should be regarded as opaque terms and always used with the `optional/1` object by passing the optional term as a parameter.

Availability:
`logtalk_load(optionals(loader))`

Author: Paulo Moura
Version: 2:2:0
Date: 2026-02-21

Compilation flags:
`static, context_switching_calls`

Provides:
type::type/1
type::check/2

Remarks:

- Type-checking support: This object also defines a type `optional` for use with the `type` library object.

Inherited public predicates:
(none)

- *Public predicates*
  - *empty/1*
  - *of/2*
  - *from_goal/3*
  - *from_goal/2*

      - *from_generator/3*
      - *from_generator/2*
      - *from_goal_or_throw/3*
      - *from_goal_or_throw/2*
    - *Protected predicates*
    - *Private predicates*
    - *Operators*

Public predicates

empty/1

Constructs an empty optional term.

Compilation flags:
`static`

Template:
`empty(Optional)`
Mode and number of proofs:
`empty(--nonvar)` - `one`

---

of/2

Constructs an optional term holding the given value.

Compilation flags:
`static`

Template:
`of(Value,Optional)`
Mode and number of proofs:
`of(@term,--nonvar)` - `one`

---

from_goal/3

Constructs an optional term holding a value bound by calling the given goal. Returns an empty optional term if the goal fails. Also returns an empty optional term if the goal throws an error, silently discarding the error information. Use `from_goal_or_throw/3` if exceptions should be propagated or the `expected` library `from_goal/3` predicate if error information should be preserved.

Compilation flags:
`static`

Template:

```
from_goal(Goal,Value,Optional)
```

Meta-predicate template:

```
from_goal(0,*,*)
```

Mode and number of proofs:

```
from_goal(+callable,--term,--nonvar) - one
```

---

from_goal/2

Constructs an optional term holding a value bound by calling the given closure. Returns an empty optional term if the closure fails. Also returns an empty optional term if the closure throws an error, silently discarding the error information. Use `from_goal_or_throw/2` if exceptions should be propagated or the `expected` library `from_goal/2` predicate if error information should be preserved.

Compilation flags:

```
static
```

Template:

```
from_goal(Closure,Optional)
```

Meta-predicate template:

```
from_goal(1,*)
```

Mode and number of proofs:

```
from_goal(+callable,--nonvar) - one
```

---

from_generator/3

Constructs optional terms with the values generated by calling the given goal. On goal failure, returns an empty optional. On goal error, also returns an empty optional, silently discarding the error information.

Compilation flags:

```
static
```

Template:

```
from_generator(Goal,Value,Optional)
```

Meta-predicate template:

```
from_generator(0,*,*)
```

Mode and number of proofs:

```
from_generator(+callable,--term,--nonvar) - one_or_more
```

---

from_generator/2

Constructs optional terms with the values generated by calling the given closure. On closure failure, returns an empty optional. On closure error, also returns an empty optional, silently discarding the error information.

Compilation flags:
```
static
```

Template:
```
from_generator(Closure,Optional)
```
Meta-predicate template:
```
from_generator(1,*)
```
Mode and number of proofs:
```
from_generator(+from_generator,--nonvar) - one_or_more
```

---

from_goal_or_throw/3

Constructs an optional term holding a value bound by calling the given goal. Returns an empty optional term if the goal fails. Propagates any exception thrown by the goal.

Compilation flags:
```
static
```

Template:
```
from_goal_or_throw(Goal,Value,Optional)
```
Meta-predicate template:
```
from_goal_or_throw(0,*,*)
```
Mode and number of proofs:
```
from_goal_or_throw(+callable,--term,--nonvar) - one_or_error
```

---

from_goal_or_throw/2

Constructs an optional term holding a value bound by calling the given closure. Returns an empty optional term if the closure fails. Propagates any exception thrown by the closure.

Compilation flags:
```
static
```

Template:
```
from_goal_or_throw(Closure,Optional)
```
Meta-predicate template:
```
from_goal_or_throw(1,*)
```
Mode and number of proofs:
```
from_goal_or_throw(+callable,--nonvar) - one_or_error
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

optional(Optional), *type*

object

1.245.3 `optional(Optional)`

Optional term handling predicates. Requires passing an optional term (constructed using the `optional` object predicates) as a parameter.

Availability:

```
logtalk_load(optionals(loader))
```

Author: Paulo Moura
Version: 1:9:0
Date: 2025-06-19

Compilation flags:

```
static, context_switching_calls
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *is_empty/0*
  - *is_present/0*
  - *if_empty/1*
  - *if_present/1*
  - *if_present_or_else/2*
  - *filter/2*

    - *map/2*
    - *flat_map/2*
    - *or/2*
    - *get/1*
    - *or_else/2*
    - *or_else_get/2*
    - *or_else_call/2*
    - *or_else_fail/1*
    - *or_else_throw/2*
    - *map_or_else/3*
    - *zip/3*
    - *flatten/1*
    - *to_expected/2*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

is_empty/0

True if the optional term is empty. See also the `if_empty/1` predicate.

Compilation flags:
`static`

Mode and number of proofs:
`is_empty` - `zero_or_one`

---

is_present/0

True if the optional term holds a value. See also the `if_present/1` predicate.

Compilation flags:
`static`

Mode and number of proofs:
`is_present` - `zero_or_one`

---

if_empty/1

Calls a goal if the optional term is empty. Succeeds otherwise.

Compilation flags:
`static`

Template:
`if_empty(Goal)`
Meta-predicate template:
`if_empty(0)`
Mode and number of proofs:
`if_empty(+callable)` - `zero_or_more`

---

if_present/1

Applies a closure to the value hold by the optional term if not empty. Succeeds otherwise.

Compilation flags:
`static`

Template:
`if_present(Closure)`
Meta-predicate template:
`if_present(1)`
Mode and number of proofs:
`if_present(+callable)` - `zero_or_more`

---

if_present_or_else/2

Applies a closure to the value hold by the optional term if not empty. Otherwise calls the given goal.

Compilation flags:
`static`

Template:
`if_present_or_else(Closure,Goal)`
Meta-predicate template:
`if_present_or_else(1,0)`
Mode and number of proofs:
`if_present_or_else(+callable,+callable)` - `zero_or_more`

---

filter/2

Returns the optional term when it is not empty and the value it holds satisfies a closure. Otherwise returns an empty optional term.

Compilation flags:
```
static
```

Template:
```
filter(Closure,NewOptional)
```
Meta-predicate template:
```
filter(1,*)
```
Mode and number of proofs:
```
filter(+callable,--nonvar) - one
```

---

map/2

When the optional term is not empty and mapping a closure with the value it holds and the new value as additional arguments is successful, returns an optional term with the new value. Otherwise returns an empty optional term.

Compilation flags:
```
static
```

Template:
```
map(Closure,NewOptional)
```
Meta-predicate template:
```
map(2,*)
```
Mode and number of proofs:
```
map(+callable,--nonvar) - one
```

---

flat_map/2

When the optional term is not empty and mapping a closure with the value it holds and the new optional term as additional arguments is successful, returns the new optional term. Otherwise returns an empty optional term.

Compilation flags:
```
static
```

Template:
```
flat_map(Closure,NewOptional)
```
Meta-predicate template:
```
flat_map(2,*)
```
Mode and number of proofs:
```
flat_map(+callable,--nonvar) - one
```

---

or/2

Returns the same optional term if not empty. Otherwise calls closure to generate a new optional term. Fails if optional term is empty and calling the closure fails or throws an error.

Compilation flags:
```
static
```

Template:
```
or(NewOptional,Closure)
```
Meta-predicate template:
```
or(*,1)
```
Mode and number of proofs:
```
or(--term,@callable) - zero_or_one
```

---

get/1

Returns the value hold by the optional term if not empty. Throws an error otherwise.

Compilation flags:
```
static
```

Template:
```
get(Value)
```
Mode and number of proofs:
```
get(--term) - one_or_error
```

Exceptions:
Optional is empty:
```
existence_error(optional_term,Optional)
```

---

or_else/2

Returns the value hold by the optional term if not empty or the given default value if the optional term is empty.

Compilation flags:
```
static
```

Template:
```
or_else(Value,Default)
```
Mode and number of proofs:
```
or_else(--term,@term) - one
```

---

or_else_get/2

Returns the value hold by the optional term if not empty. Applies a closure to compute the value otherwise. Throws an error when the optional term is empty and the value cannot be computed.

Compilation flags:
`static`

Template:
`or_else_get(Value,Closure)`
Meta-predicate template:
`or_else_get(*,1)`
Mode and number of proofs:
`or_else_get(--term,+callable)` - `one_or_error`

Exceptions:
Optional is empty and the term cannot be computed:
`existence_error(optional_term,Optional)`

---

or_else_call/2

Returns the value hold by the optional term if not empty or calls a goal deterministically if the optional term is empty.

Compilation flags:
`static`

Template:
`or_else_call(Value,Goal)`
Meta-predicate template:
`or_else_call(*,0)`
Mode and number of proofs:
`or_else_call(--term,+callable)` - `zero_or_one`

---

or_else_fail/1

Returns the value hold by the optional term if not empty. Fails otherwise. Usually called to skip over empty optional terms.

Compilation flags:
`static`

Template:
`or_else_fail(Value)`
Mode and number of proofs:
`or_else_fail(--term)` - `zero_or_one`

---

or_else_throw/2

Returns the value hold by the optional term if not empty. Throws the given error otherwise.

Compilation flags:
```
static
```

Template:
```
or_else_throw(Value,Error)
```
Mode and number of proofs:
```
or_else_throw(--term,@nonvar) - one_or_error
```

---

map_or_else/3

When the optional term is not empty and mapping a closure with the value it holds and the new value as additional arguments is successful, returns the new value. Otherwise returns the given default value.

Compilation flags:
```
static
```

Template:
```
map_or_else(Closure,Default,Value)
```
Meta-predicate template:
```
map_or_else(2,*,*)
```
Mode and number of proofs:
```
map_or_else(+callable,@term,--term) - one
```

---

zip/3

When both this optional and the other optional hold values and applying a closure with both values and the new value as additional arguments is successful, returns an optional term with the new value. Otherwise returns an empty optional term.

Compilation flags:
```
static
```

Template:
```
zip(Closure,OtherOptional,NewOptional)
```
Meta-predicate template:
```
zip(3,*,*)
```
Mode and number of proofs:
```
zip(+callable,+nonvar,--nonvar) - one
```

---

flatten/1

Flattens a nested optional term. When the optional term holds a value that is itself an optional term, returns the inner optional term. When the optional term holds a non-optional value, returns the same optional term. When the optional term is empty, returns an empty optional term.

Compilation flags:

```
static
```

Template:

```
flatten(NewOptional)
```

Mode and number of proofs:

```
flatten(--nonvar) - one
```

---

to_expected/2

Converts the optional term to an expected term. Returns an expected term holding the value if the optional term is not empty. Returns an expected term with the given error otherwise.

Compilation flags:

```
static
```

Template:

```
to_expected(Error,Expected)
```

Mode and number of proofs:

```
to_expected(@term,--nonvar) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

optional

1.246 options

category

1.246.1 `options`

Options processing predicates. Options are represented by compound terms where the functor is the option name.

Availability:
`logtalk_load(options(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2026-09-08

Compilation flags:
`static`

Implements:
`public` *options_protocol*
Uses:
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.246.2 `options_protocol`

Options protocol.

Availability:
`logtalk_load(options(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2022-01-03

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *check_option/1*
  - *check_options/1*
  - *valid_option/1*
  - *valid_options/1*
  - *default_option/1*
  - *default_options/1*
  - *option/2*
  - *option/3*
- *Protected predicates*
  - *merge_options/2*
  - *fix_options/2*
  - *fix_option/2*
- *Private predicates*
- *Operators*

Public predicates

check_option/1

Succeeds if the option is valid. Throws an error otherwise.

Compilation flags:
`static`

Template:
`check_option(Option)`
Mode and number of proofs:
`check_option(@term)` - `one_or_error`

Exceptions:
`Option` is a variable:
`instantiation_error`
`Option` is neither a variable nor a compound term:
`type_error(compound,Option)`
`Option` is a compound term but not a valid option:
`domain_error(option,Option)`

---

check_options/1

Succeeds if all the options in a list are valid. Throws an error otherwise.

Compilation flags:
`static`

Template:
`check_options(Options)`
Mode and number of proofs:
`check_options(@term)` - `one_or_error`

Exceptions:
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

valid_option/1

Succeeds if the option is valid.

Compilation flags:
`static`

Template:
`valid_option(Option)`
Mode and number of proofs:
`valid_option(@term)` - `zero_or_one`

---

valid_options/1

Succeeds if all the options in a list are valid.

Compilation flags:
`static`

Template:
`valid_options(Options)`
Mode and number of proofs:
`valid_options(@term)` - `one`

---

default_option/1

Enumerates, by backtracking, the default options.

Compilation flags:
`static`

Template:
`default_option(Option)`
Mode and number of proofs:
`default_option(?compound)` - `zero_or_more`

---

default_options/1

Returns a list of the default options.

Compilation flags:
`static`

Template:
`default_options(Options)`
Mode and number of proofs:
`default_options(-list(compound))` - `one`

---

option/2

True iff `Option` unifies with the first occurrence of the same option in the `Options` list.

Compilation flags:
`static`

Template:
`option(Option,Options)`
Mode and number of proofs:
`option(+compound,+list(compound))` - `zero_or_one`

---

option/3

True iff `Option` unifies with the first occurrence of the same option in the `Options` list or, when that is not the case, if `Option` unifies with `Default`.

Compilation flags:
`static`

Template:
`option(Option,Options,Default)`
Mode and number of proofs:
`option(+compound,+list(compound),+compound)` - `zero_or_one`

Protected predicates

merge_options/2

Merges the user options with the default options, returning the final list of options. Calls the `fix_options/2` predicate to preprocess the options after merging. Callers must ensure, if required, that the user options are valid.

Compilation flags:

```
static
```

Template:

```
merge_options(UserOptions,Options)
```

Mode and number of proofs:

```
merge_options(+list(compound),-list(compound)) - one
```

---

fix_options/2

Fixes a list of options, returning the list of options.

Compilation flags:

```
static
```

Template:

```
fix_options(Options,FixedOptions)
```

Mode and number of proofs:

```
fix_options(+list(compound),-list(compound)) - one
```

---

fix_option/2

Fixes an option.

Compilation flags:

```
static
```

Template:

```
fix_option(Option,FixedOption)
```

Mode and number of proofs:

```
fix_option(+compound,-compound) - zero_or_one
```

Private predicates

(none)

Operators

(none)

See also

options

1.247 os

object

1.247.1 os

Portable operating-system access predicates.

Availability:
`logtalk_load(os(loader))`

Author: Paulo Moura
Version: 1:105:3
Date: 2026-11-13

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *osp*
Uses:
atom
list
Aliases:
`osp` `absolute_file_name/2` as `expand_path/2`

Remarks:

- File path expansion: To ensure portability, all file paths are expanded before being handed to the backend Prolog system.
- Exception terms: Currently, there is no standardization of the exception terms thrown by the different backend Prolog systems.
- B-Prolog portability: The `wall_time/1` predicate is not supported.
- CxProlog portability: The `date_time/7` predicate returns zeros for all arguments.
- JIProlog portability: The `file_permission/2` and `command_line_arguments/1` predicates are not supported.
- Quintus Prolog: The `pid/1` and `shell/2` predicates are not supported.
- XSB portability: The `command_line_arguments/1` predicate is not supported.

Inherited public predicates:

absolute_file_name/2 *change_directory/1* *command_line_arguments/1* *copy_file/2* *cpu_time/1* *date_time/7* *decompose_file_name/3* *decompose_file_name/4* *delete_directory/1* *delete_directory_and_contents/1* *delete_directory_contents/1* *delete_file/1* *directory_exists/1* *directory_files/2* *directory_files/3* *ensure_directory/1* *ensure_file/1* *environment_variable/2* *file_exists/1* *file_modification_time/2* *file_permission/2* *file_size/2* *full_device_path/1* *internal_os_path/2* *is_absolute_file_name/1* *make_directory/1* *make_directory_path/1* *null_device_path/1* *operating_system_machine/1* *operating_system_name/1* *operating_system_release/1* *operating_system_type/1* *path_concat/3* *pid/1* *read_only_device_path/1* *rename_file/2* *resolve_command_path/2* *shell/1* *shell/2* *sleep/1* *temporary_directory/1* *time_stamp/1* *wall_time/1* *working_directory/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

os_types

category

1.247.2 `os_types`

A set of operating-system related types.

Availability:
`logtalk_load(os(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2021-02-12

Compilation flags:
`static`

Provides:

type::type/1
type::check/2

Uses:

list
os

Remarks:

- Provided types: This category adds `file`, `file(Extensions)`, `file(Extensions,Permissions)`, `directory`, `directory(Permissions)`, and `environment_variable` types for type-checking when using the `type` library object.
- Type `file`: For checking if a term is an atom and an existing file.
- Type `file(Extensions)`: For checking if a term is an atom and an existing file with one of the listed extensions (specified as `'.ext'`).
- Type `file(Extensions,Permissions)`: For checking if a term is an atom and an existing file with one of the listed extensions (specified as `'.ext'`) and listed permissions ({`read`, `write`, `execute`}).
- Type `directory`: For checking if a term is an atom and an existing directory.
- Type `directory(Permissions)`: For checking if a term is an atom and an existing directory with the listed permissions ({`read`, `write`, `execute`}).
- Type `environment_variable`: For checking if a term is an atom and an existing environment variable.

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

osp, *os*, *type*

protocol

1.247.3 osp

Portable operating-system access protocol.

Availability:

```
logtalk_load(os(loader))
```

Author: Paulo Moura
Version: 1:44:0
Date: 2026-06-10

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

- Error handling: Predicates that require a file or directory to exist throw an error when that is not the case. But the exact exception term is currently backend Prolog compiler dependent.
- CPU and wall time accuracy: Depends on the backend and can be different between CPU and wall time (e.g. CPU time can have nanosecond accuracy with wall time only having millisecond accuracy).

Inherited public predicates:

(none)

- *Public predicates*
  - *pid/1*
  - *shell/2*
  - *shell/1*
  - *is_absolute_file_name/1*
  - *absolute_file_name/2*
  - *decompose_file_name/3*
  - *decompose_file_name/4*
  - *path_concat/3*
  - *internal_os_path/2*
  - *make_directory/1*
  - *make_directory_path/1*

      - *delete_directory/1*
      - *delete_directory_contents/1*
      - *delete_directory_and_contents/1*
      - *change_directory/1*
      - *working_directory/1*
      - *temporary_directory/1*
      - *null_device_path/1*
      - *full_device_path/1*
      - *read_only_device_path/1*
      - *resolve_command_path/2*
      - *directory_files/2*
      - *directory_files/3*
      - *directory_exists/1*
      - *ensure_directory/1*
      - *file_exists/1*
      - *file_modification_time/2*
      - *file_size/2*
      - *file_permission/2*
      - *copy_file/2*
      - *rename_file/2*
      - *delete_file/1*
      - *ensure_file/1*
      - *environment_variable/2*
      - *time_stamp/1*
      - *date_time/7*
      - *cpu_time/1*
      - *wall_time/1*
      - *operating_system_type/1*
      - *operating_system_name/1*
      - *operating_system_machine/1*
      - *operating_system_release/1*
      - *command_line_arguments/1*
      - *sleep/1*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

pid/1

Returns the process identifier of the running process.

Compilation flags:

```
static
```

Template:

```
pid(PID)
```

Mode and number of proofs:
`pid(-integer)` - `one`

---

shell/2

Runs an operating-system shell command and returns its exit status.

Compilation flags:
`static`

Template:
`shell(Command,Status)`
Mode and number of proofs:
`shell(+atom,-integer)` - `one`

---

shell/1

Runs an operating-system shell command.

Compilation flags:
`static`

Template:
`shell(Command)`
Mode and number of proofs:
`shell(+atom)` - `zero_or_one`

---

is_absolute_file_name/1

True iff the argument is an absolute file path. On POSIX systems, this predicate is true if `File` starts with a `/`. On Windows systems, this predicate is true if `File` starts with a drive letter. No attempt is made to expand `File` as a path.

Compilation flags:
`static`

Template:
`is_absolute_file_name(File)`
Mode and number of proofs:
`is_absolute_file_name(+atom)` - `zero_or_one`

---

absolute_file_name/2

Expands a file name to an absolute file path. An environment variable at the beginning of the file name is also expanded. A slash at the end of `File` is preserved in `Path`.

Compilation flags:

```
static
```

Template:

```
absolute_file_name(File,Path)
```

Mode and number of proofs:

```
absolute_file_name(+atom,-atom) - one
```

---

decompose_file_name/3

Decomposes a file name into its directory (which always ends with a slash; `./` is returned if absent) and its basename (which can be the empty atom).

Compilation flags:

```
static
```

Template:

```
decompose_file_name(File,Directory,Basename)
```

Mode and number of proofs:

```
decompose_file_name(+atom,?atom,?atom) - one
```

---

decompose_file_name/4

Decomposes a file name into its directory (which always ends with a slash; `./` is returned if absent), name (that can be the empty atom), and extension (which starts with a `.` when defined; the empty atom otherwise).

Compilation flags:

```
static
```

Template:

```
decompose_file_name(File,Directory,Name,Extension)
```

Mode and number of proofs:

```
decompose_file_name(+atom,?atom,?atom,?atom) - one
```

---

path_concat/3

Concatenates a path prefix and a path suffix, adding a `/` separator if required. Returns `Suffix` when it is an absolute path. Returns `Prefix` with a trailing `/` appended if missing when `Suffix` is the empty atom.

Compilation flags:
`static`

Template:
`path_concat(Prefix,Suffix,Path)`
Mode and number of proofs:
`path_concat(+atom,+atom,--atom)` - `one`

---

internal_os_path/2

Converts between the internal path representation (which is backend dependent) and the operating-system native path representation.

Compilation flags:
`static`

Template:
`internal_os_path(InternalPath,OSPath)`
Mode and number of proofs:
`internal_os_path(+atom,-atom)` - `one`
`internal_os_path(-atom,+atom)` - `one`

---

make_directory/1

Makes a new directory. Succeeds if the directory already exists.

Compilation flags:
`static`

Template:
`make_directory(Directory)`
Mode and number of proofs:
`make_directory(+atom)` - `one`

---

make_directory_path/1

Makes a new directory creating all the intermediate directories if necessary. Succeeds if the directory already exists.

Compilation flags:
```
static
```

Template:
```
make_directory_path(Directory)
```
Mode and number of proofs:
```
make_directory_path(+atom) - one
```

---

delete_directory/1

Deletes an empty directory. Throws an error if the directory does not exist.

Compilation flags:
```
static
```

Template:
```
delete_directory(Directory)
```
Mode and number of proofs:
```
delete_directory(+atom) - one_or_error
```

---

delete_directory_contents/1

Deletes directory contents. Throws an error if the directory does not exist.

Compilation flags:
```
static
```

Template:
```
delete_directory_contents(Directory)
```
Mode and number of proofs:
```
delete_directory_contents(+atom) - one_or_error
```

---

delete_directory_and_contents/1

Deletes directory and its contents. Throws an error if the directory does not exist.

Compilation flags:
`static`

Template:
`delete_directory_and_contents(Directory)`
Mode and number of proofs:
`delete_directory_and_contents(+atom)` - `one_or_error`

---

change_directory/1

Changes current working directory. Throws an error if the directory does not exist.

Compilation flags:
`static`

Template:
`change_directory(Directory)`
Mode and number of proofs:
`change_directory(+atom)` - `one_or_error`

---

working_directory/1

Current working directory.

Compilation flags:
`static`

Template:
`working_directory(Directory)`
Mode and number of proofs:
`working_directory(?atom)` - `zero_or_one`

---

temporary_directory/1

Temporary directory. Tries first environment variables: `TEMP` and `TMP` on Windows systems; `TMPDIR` on POSIX systems. When not defined, tries default locations. Returns the working directory as last resort.

Compilation flags:
`static`

Template:
`temporary_directory(Directory)`
Mode and number of proofs:
`temporary_directory(?atom)` - `one`

---

null_device_path/1

Null device path: `nul` on Windows systems and `/dev/null` on POSIX systems.

Compilation flags:
`static`

Template:
`null_device_path(Path)`
Mode and number of proofs:
`null_device_path(?atom)` - `one`

---

full_device_path/1

Full device path: `/dev/full` on Linux and BSD systems. Fails on other systems. Experimental.

Compilation flags:
`static`

Template:
`full_device_path(Path)`
Mode and number of proofs:
`full_device_path(?atom)` - `zero_or_one`

---

read_only_device_path/1

Read-only device path: `/dev/urandom` on macOS. Fails on other systems. Experimental.

Compilation flags:
`static`

Template:
`read_only_device_path(Path)`
Mode and number of proofs:
`read_only_device_path(?atom)` - `zero_or_one`

---

resolve_command_path/2

Resolves a command path given its name.

Compilation flags:
`static`

Template:
`resolve_command_path(Command,Path)`
Mode and number of proofs:
`resolve_command_path(+atom,-atom)` - `zero_or_one`

---

directory_files/2

Returns a list of all files (including directories, regular files, and hidden directories and files) in a directory. File paths are relative to the directory. Throws an error if the directory does not exist.

Compilation flags:
`static`

Template:
`directory_files(Directory,Files)`
Mode and number of proofs:
`directory_files(+atom,-list(atom))` - `one_or_error`

---

directory_files/3

Returns a list of files filtered using the given list of options. Invalid options are ignored. Default option values are equivalent to `directory_files/2`. Throws an error if the directory does not exist.

Compilation flags:

```
static
```

Template:

```
directory_files(Directory,Files,Options)
```

Mode and number of proofs:

```
directory_files(+atom,-list(atom),+list(compound)) - one_or_error
```

Remarks:

- Option `paths/1`: Possible values are `relative` and `absolute`. Default is `relative`.
- Option `type/1`: Possible values are `all`, `regular`, `directory`. Default is `all`.
- Option `extensions/1`: Argument is a list of required extensions (using the format `'.ext'`). Default is the empty list.
- Option `prefixes/1`: Argument is a list of required file prefixes (atoms). Default is the empty list.
- Option `suffixes/1`: Argument is a list of required file suffixes (atoms). Default is the empty list.
- Option `dot_files/1`: Possible values are `true` and `false`. Default is `true`.

---

directory_exists/1

True if the specified directory exists (irrespective of directory permissions).

Compilation flags:

```
static
```

Template:

```
directory_exists(Directory)
```

Mode and number of proofs:

```
directory_exists(+atom) - zero_or_one
```

---

ensure_directory/1

Ensures that a directory exists, creating it if necessary.

Compilation flags:

```
static
```

Template:

```
ensure_directory(Directory)
```

Mode and number of proofs:

```
ensure_directory(+atom) - one
```

---

file_exists/1

True if the specified file exists and is a regular file (irrespective of file permissions).

Compilation flags:
`static`

Template:
`file_exists(File)`
Mode and number of proofs:
`file_exists(+atom)` - `zero_or_one`

---

file_modification_time/2

File modification time (which can be used for comparison). Throws an error if the file does not exist.

Compilation flags:
`static`

Template:
`file_modification_time(File,Time)`
Mode and number of proofs:
`file_modification_time(+atom,-integer)` - `one_or_error`

---

file_size/2

File size (in bytes). Throws an error if the file does not exist.

Compilation flags:
`static`

Template:
`file_size(File,Size)`
Mode and number of proofs:
`file_size(+atom,-integer)` - `one_or_error`

---

file_permission/2

True iff the specified file has the specified permission (`read`, `write`, or `execute`). Throws an error if the file does not exist.

Compilation flags:

```
static
```

Template:

```
file_permission(File,Permission)
```

Mode and number of proofs:

```
file_permission(+atom,+atom) - zero_or_one_or_error
```

---

copy_file/2

Copies a file. Throws an error if the original file does not exist or if the copy cannot be created.

Compilation flags:

```
static
```

Template:

```
copy_file(File,Copy)
```

Mode and number of proofs:

```
copy_file(+atom,+atom) - one_or_error
```

---

rename_file/2

Renames a file or a directory. Throws an error if the file or directory does not exist.

Compilation flags:

```
static
```

Template:

```
rename_file(Old,New)
```

Mode and number of proofs:

```
rename_file(+atom,+atom) - one_or_error
```

---

delete_file/1

Deletes a file. Throws an error if the file does not exist.

Compilation flags:
`static`

Template:
`delete_file(File)`
Mode and number of proofs:
`delete_file(+atom)` - `one_or_error`

---

ensure_file/1

Ensures that a file exists, creating it if necessary.

Compilation flags:
`static`

Template:
`ensure_file(File)`
Mode and number of proofs:
`ensure_file(+atom)` - `one`

---

environment_variable/2

Returns an environment variable value. Fails if the variable does not exists.

Compilation flags:
`static`

Template:
`environment_variable(Variable,Value)`
Mode and number of proofs:
`environment_variable(+atom,?atom)` - `zero_or_one`

---

time_stamp/1

Returns a system-dependent time stamp, which can be used for sorting, but should be regarded otherwise as an opaque term.

Compilation flags:
`static`

Template:
`time_stamp(Time)`
Mode and number of proofs:
`time_stamp(-ground)` - `one`

---

date_time/7

Returns the current date and time. Note that most backends do not provide sub-second accuracy and in those cases the value of the `Milliseconds` argument is always zero.

Compilation flags:
`static`

Template:
`date_time(Year,Month,Day,Hours,Minutes,Seconds,Milliseconds)`
Mode and number of proofs:
`date_time(-integer,-integer,-integer,-integer,-integer,-integer,-integer)` - `one`

---

cpu_time/1

System cpu time in seconds. Accuracy depends on the backend.

Compilation flags:
`static`

Template:
`cpu_time(Seconds)`
Mode and number of proofs:
`cpu_time(-number)` - `one`

---

wall_time/1

Wall time in seconds. Accuracy depends on the backend.

Compilation flags:
`static`

Template:
`wall_time(Seconds)`
Mode and number of proofs:
`wall_time(-number)` - `one`

---

operating_system_type/1

Operating system type. Possible values are `unix`, `windows`, and `unknown`.

Compilation flags:
`static`

Template:
`operating_system_type(Type)`
Mode and number of proofs:
`operating_system_type(?atom)` - `zero_or_one`

---

operating_system_name/1

Operating system name. On POSIX systems, it returns the value of `uname -s`. On macOS systems, it returns `'Darwin'`. On Windows systems, it returns `'Windows'`.

Compilation flags:
`static`

Template:
`operating_system_name(Name)`
Mode and number of proofs:
`operating_system_name(?atom)` - `zero_or_one`

---

operating_system_machine/1

Operating system hardware platform. On POSIX systems, it returns the value of `uname -m`. On Windows systems, it returns the value of the `PROCESSOR_ARCHITECTURE` environment variable.

Compilation flags:
`static`

Template:
`operating_system_machine(Machine)`
Mode and number of proofs:
`operating_system_machine(?atom) - zero_or_one`

---

operating_system_release/1

Operating system release. On POSIX systems, it returns the value of `uname -r`. On Windows systems, it uses `WMI` code.

Compilation flags:
`static`

Template:
`operating_system_release(Release)`
Mode and number of proofs:
`operating_system_release(?atom) - zero_or_one`

---

command_line_arguments/1

Returns a list with the command line arguments that occur after `--`.

Compilation flags:
`static`

Template:
`command_line_arguments(Arguments)`
Mode and number of proofs:
`command_line_arguments(-list(atom)) - one`

`sleep/1`

Suspends execution the given number of seconds.

Compilation flags:
`static`

Template:
`sleep(Seconds)`
Mode and number of proofs:
`sleep(+number)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

os, *os_types*

1.248 otp

object

1.248.1 `otp`

HOTP and TOTP generation and verification predicates.

Availability:
`logtalk_load(otp(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-25

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *otp_protocol*

Uses:

base32
hmac
list

Remarks:

(none)

Inherited public predicates:

hotp/5 *hotp_verify/7* *totp/5* *totp_verify/7*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

base32, *hmac*, *sha1*, *sha256*, *sha512_256*

protocol

1.248.2 `otp_protocol`

Protocol for HOTP and TOTP generation and verification as specified in RFC 4226 and RFC 6238.

Availability:

`logtalk_load(otp(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:

`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *hotp/5*
  - *totp/5*
  - *hotp_verify/7*
  - *totp_verify/7*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

hotp/5

Computes an HOTP value for the given hash object, secret, moving counter, and number of digits. The secret can be either a raw byte list or a Base32 wrapper term of the form `base32(atom(Atom))`, `base32(chars(Chars))`, or `base32(codes(Codes))`.

Compilation flags:
`static`

Template:
`hotp(Hash,Secret,Counter,Digits,OTP)`

Mode and number of proofs:
`hotp(+object_identifier,+term,+integer,+integer,--atom)` - `one`

---

totp/5

Computes a TOTP value using the standard 30-second time step and Unix epoch `T0 = 0` for the given hash object, secret, Unix time, and number of digits.

Compilation flags:
`static`

Template:

`totp(Hash,Secret,UnixTime,Digits,OTP)`

Mode and number of proofs:

`totp(+object_identifier,+term,+integer,+integer,--atom)` - `one`

`hotp_verify/7`

Verifies an HOTP value by searching from the given counter through the bounded forward counter window. Returns the matched counter on success.

Compilation flags:

`static`

Template:

`hotp_verify(Hash,Secret,Counter,Window,Digits,OTP,MatchedCounter)`

Mode and number of proofs:

`hotp_verify(+object_identifier,+term,+integer,+integer,+integer,+atom,--integer)` - `zero_or_one_or_error`

Exceptions:

`Hash`, `Secret`, `Counter`, `Window`, `Digits`, or `OTP` is a variable or a partial list:

`instantiation_error`

`Hash` is not a supported OTP hash object:

`domain_error(otp_hash,Hash)`

`Secret` is neither a partial list nor a list:

`type_error(list,Secret)`

An element `Byte` of the *Secret*` list is neither a variable nor an integer:

`type_error(integer,Byte)`

An element `Byte` of the `Secret` list is an integer but not a valid byte:

`domain_error(byte,Byte)`

`Secret` is not a valid Base32 source:

`domain_error(base32_source,Secret)`

`Counter` or `Window` is neither a variable nor an integer:

`type_error(integer,Integer)`

`Counter` or `Window` is an integer but not a non-negative integer:

`domain_error(non_negative_integer,Integer)`

`Digits` is neither a variable nor an integer:

`type_error(integer,Digits)`

`Digits` is an integer but not a positive integer:

`domain_error(positive_integer,Digits)`

`OTP` neither a variable nor an atom:

`type_error(atom,OTP)`

`OTP` does not contain exactly `Digits` decimal digits:

`domain_error(otp_value,OTP)`

A searched counter exceeds the 64-bit HOTP moving-factor range:

`domain_error(otp_moving_factor,Counter)`

---

totp_verify/7

Verifies a TOTP value by searching within the bounded symmetric time-step window around the current time step. Returns the matched time step on success.

Compilation flags:

`static`

Template:

`totp_verify(Hash,Secret,UnixTime,Window,Digits,OTP,MatchedTimeStep)`

Mode and number of proofs:

`totp_verify(+object_identifier,+term,+integer,+integer,+integer,+atom,--integer)` - `zero_or_one_or_error`

Exceptions:

`Hash`, `Secret`, `UnixTime`, `Window`, `Digits`, or `OTP` is a variable or a partial list:

`instantiation_error`

`Hash` is not a supported OTP hash object:

`domain_error(otp_hash,Hash)`

`Secret` is neither a partial list nor a list:

`type_error(list,Secret)`

An element `Byte` of the *Secret*` list is neither a variable nor an integer:

`type_error(integer,Byte)`

An element `Byte` of the `Secret` list is an integer but not a valid byte:

`domain_error(byte,Byte)`

`Secret` is not a valid Base32 source:

`domain_error(base32_source,Secret)`

`UnixTime` or `Window` is neither a variable nor an integer:

`type_error(integer,Integer)`

`UnixTime` or `Window` is an integer but not a non-negative integer:

`domain_error(non_negative_integer,Integer)`

`Digits` is neither a variable nor an integer:

`type_error(integer,Digits)`

`Digits` is an integer but not a positive integer:

`domain_error(positive_integer,Digits)`

`OTP` is neither a variable nor an atom:

`type_error(atom,OTP)`

`OTP` does not contain exactly `Digits` decimal digits:

`domain_error(otp_value,OTP)`

A searched time step exceeds the 64-bit HOTP moving-factor range:

`domain_error(otp_moving_factor,TimeStep)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

hmac_protocol

1.249 packs

protocol

1.249.1 `pack_protocol`

Pack specification protocol. Objects implementing this protocol should be named after the pack with a `_pack` suffix and saved in a file with the same name as the object.

Availability:
`logtalk_load(packs(loader))`

Author: Paulo Moura
Version: 0:18:0
Date: 2025-05-21

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *name/1*
  - *description/1*
  - *license/1*

    - *home/1*
    - *version/6*
    - *note/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

name/1

Pack name.

Compilation flags:
`static`

Template:
`name(Name)`
Mode and number of proofs:
`name(?atom)` - `zero_or_one`

---

description/1

Pack one line description.

Compilation flags:
`static`

Template:
`description(Description)`
Mode and number of proofs:
`description(?atom)` - `zero_or_one`

---

license/1

Pack license. Specified using the identifier from the SPDX License List (https://spdx.org/licenses/) when possible.

Compilation flags:
`static`

Template:
`license(License)`
Mode and number of proofs:
`license(?atom)` - `zero_or_one`

---

home/1

Pack home HTTPS or file URL.

Compilation flags:

```
static
```

Template:

```
home(Home)
```

Mode and number of proofs:

```
home(?atom) - zero_or_one
```

---

version/6

Table of available versions.

Compilation flags:

```
static
```

Template:

```
version(Version,Status,URL,Checksum,Dependencies,Portability)
```

Mode and number of proofs:

```
version(?compound,?atom,-atom,-pair(atom,atom),-list(pair(atom,callable)),?atom) -
zero_or_more
version(?compound,?atom,-atom,-pair(atom,atom),-list(pair(atom,callable)),
-list(atom)) - zero_or_more
```

Remarks:

- Version: This argument uses the same format as entity versions: `Major:Minor:Patch`. Semantic versioning should be used.
- Status: Version development status: `stable`, `rc`, `beta`, `alpha`, `experimental`, or `deprecated`.
- URL: File URL for a local directory, file URL for a local archive, download HTTPS URL for the pack archive, or download git archive URL for the pack archive.
- Checksum: A pair where the key is the hash algorithm and the value is the checksum. Currently, the hash algorithm must be `sha256`. For `file://` URLs of local directories, use `none` instead of a pair.
- Dependencies: Pack dependencies list. Each dependency is a `Dependency Operator Version` term. `Operator` is a term comparison operator. Valid `Dependency` values are `Registry::Pack`, `os(Name,Machine)`, `logtalk`, and a backend identifier atom.
- Portability: Either the atom `all` or a list of the supported backend Prolog compilers (using the identifier atoms used by the `prolog_dialect` flag).
- Clause order: Versions must be listed ordered from newest to oldest.

`note/3`

Table of notes per action and version.

Compilation flags:
`static`

Template:
`note(Action,Version,Note)`
Mode and number of proofs:
`note(?atom,?term,-atom)` - `zero_or_more`

Remarks:

- Action: Possible values are `install`, `update`, and `uninstall`. When unbound, the note apply to all actions.
- Version: Version being installed, updated, or uninstalled. When unbound, the note apply to all versions.
- Note: Note to print when performing an action on a pack version.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.249.2 `packs`

Pack handling predicates.

Availability:
`logtalk_load(packs(loader))`

Author: Paulo Moura
Version: 0:92:0
Date: 2026-10-28

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *packs_common*
`public` *options*
Uses:
git

list
logtalk
os
registries
type
user

Remarks:
(none)

Inherited public predicates:
check_option/1 check_options/1 default_option/1 default_options/1 directory/1 directory/2 help/0 logtalk_packs/0 logtalk_packs/1 option/2 option/3 pin/0 pin/1 pinned/1 prefix/0 prefix/1 readme/1 readme/2 reset/0 setup/0 unpin/0 unpin/1 valid_option/1 valid_options/1 verify_commands_availability/0

- *Public predicates*
  - *available/2*
  - *available/1*
  - *available/0*
  - *installed/4*
  - *installed/3*
  - *installed/1*
  - *installed/0*
  - *outdated/5*
  - *outdated/4*
  - *outdated/2*
  - *outdated/1*
  - *outdated/0*
  - *orphaned/2*
  - *orphaned/0*
  - *versions/3*
  - *describe/2*
  - *describe/1*
  - *pack_metadata/4*
  - *pack_property/4*
  - *pack_object/3*
  - *loaded_pack/3*
  - *loaded_pack_file/4*
  - *search/1*
  - *pack_dependency/6*
  - *loaded_pack_dependency/6*
  - *install/4*
  - *install/3*
  - *install/2*
  - *install/1*
  - *update/3*
  - *update/2*

      - *update/1*
      - *update/0*
      - *uninstall/2*
      - *uninstall/1*
      - *uninstall/0*
      - *clean/2*
      - *clean/1*
      - *clean/0*
      - *save/2*
      - *save/1*
      - *restore/2*
      - *restore/1*
      - *dependents/3*
      - *dependents/2*
      - *dependents/1*
      - *lint/2*
      - *lint/1*
      - *lint/0*
  - *Protected predicates*
  - *Private predicates*
      - *save_setup_terms/2*
      - *lock_registry_facts/3*
      - *lock_pack_facts/2*
      - *check_lock_options/1*
  - *Operators*

Public predicates

available/2

Enumerates, by backtracking, all available packs.

Compilation flags:
    `static`

Template:
    `available(Registry,Pack)`
Mode and number of proofs:
    `available(?atom,?atom)` - `zero_or_more`

Exceptions:
    `Registry` is neither a variable nor an atom:
        `type_error(atom,Registry)`
    `Pack` is neither a variable nor an atom:
        `type_error(atom,Pack)`

available/1

Lists all the packs that are available for installation from the given registry.

Compilation flags:
`static`

Template:
`available(Registry)`
Mode and number of proofs:
`available(+atom)` - `one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`

---

available/0

Lists all the packs that are available for installation from all defined registries.

Compilation flags:
`static`

Mode and number of proofs:
`available` - `one`

---

installed/4

Enumerates by backtracking all installed packs.

Compilation flags:
`static`

Template:
`installed(Registry,Pack,Version,Pinned)`
Mode and number of proofs:
`installed(?atom,?atom,?compound,?boolean)` - `zero_or_more`

Exceptions:
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

`Version` is neither a variable nor a compound term:
`type_error(compound,Version)`
`Pinned` is neither a variable nor a boolean:
`type_error(boolean,Pinned)`

`installed/3`

Enumerates by backtracking all installed packs.

Compilation flags:
`static`

Template:
`installed(Registry,Pack,Version)`
Mode and number of proofs:
`installed(?atom,?atom,?compound)` - `zero_or_more`

Exceptions:
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`
`Version` is neither a variable nor a compound term:
`type_error(compound,Version)`

`installed/1`

Lists all the packs that are installed from the given registry. Fails if the registry is unknown.

Compilation flags:
`static`

Template:
`installed(Registry)`
Mode and number of proofs:
`installed(+atom)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`

---

installed/0

Lists all the packs that are installed.

Compilation flags:
`static`

Mode and number of proofs:
`installed` - `one`

---

outdated/5

Enumerates by backtracking all installed but outdated packs (together with the current version installed and the latest version available) using the given options.

Compilation flags:
`static`

Template:
`outdated(Registry,Pack,Version,LatestVersion,Options)`

Mode and number of proofs:
`outdated(?atom,?atom,?compound,?compound,++list(compound))` - `zero_or_more`

Exceptions:
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`
`Version` is neither a variable nor a compound term:
`type_error(compound,Version)`
`LatestVersion` is neither a variable nor a compound term:
`type_error(compound,LatestVersion)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

Remarks:

- `compatible(Boolean)` option: Restrict listing to compatible packs. Default is `true`.
- `status(Status)` option: Restrict listing to updates with the given status. Default is `[stable,rc,beta,alpha]`. Set to `all` to also list `experimental` and `deprecated` updates.

---

outdated/4

Enumerates by backtracking all installed but outdated packs (together with the current version installed and the latest version available) using default options.

Compilation flags:
`static`

Template:
`outdated(Registry,Pack,Version,LatestVersion)`
Mode and number of proofs:
`outdated(?atom,?atom,?compound,?compound)` - `zero_or_more`

Exceptions:
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`
`Version` is neither a variable nor a compound term:
`type_error(compound,Version)`
`LatestVersion` is neither a variable nor a compound term:
`type_error(compound,LatestVersion)`

See also:
outdated/5

---

outdated/2

Lists all the packs from the given registry that are installed but outdated using the given options.

Compilation flags:
`static`

Template:
`outdated(Registry,Options)`
Mode and number of proofs:
`outdated(+atom,++list(compound))` - `one`

Exceptions:
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

Remarks:

- `compatible(Boolean)` option: Restrict installation to compatible packs. Default is `true`.
- `status(Status)` option: Restrict listing to updates with the given status. Default is `[stable,rc,beta,alpha]`. Set to `all` to also list `experimental` and `deprecated` updates.

---

outdated/1

Lists all the packs from the given registry that are installed but outdated using default options.

Compilation flags:
`static`

Template:
`outdated(Registry)`
Mode and number of proofs:
`outdated(+atom)` - `one`

Exceptions:
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`

See also:
outdated/2

---

outdated/0

Lists all the packs that are installed but outdated using default options.

Compilation flags:
`static`

Mode and number of proofs:
`outdated` - `one`

See also:
outdated/1

---

orphaned/2

Lists all the packs that are installed but whose registry is no longer defined.

Compilation flags:
`static`

Template:
`orphaned(Registry,Pack)`
Mode and number of proofs:
`orphaned(?atom,?atom)` - `zero_or_more`

Exceptions:
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

---

orphaned/0

Lists all the packs that are installed but whose registry is no longer defined.

Compilation flags:
`static`

Mode and number of proofs:
`orphaned` - `one`

---

versions/3

Returns a list of all available pack versions. Fails if the pack is unknown.

Compilation flags:
`static`

Template:
`versions(Registry,Pack,Versions)`
Mode and number of proofs:
`versions(+atom,+atom,-list)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`

`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

---

describe/2

Describes a registered pack, including installed version if applicable. Fails if the pack is unknown.

Compilation flags:
`static`

Template:
`describe(Registry,Pack)`
Mode and number of proofs:
`describe(+atom,+atom)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

---

describe/1

Describes a registered pack, including installed version if applicable. Fails if the pack is unknown.

Compilation flags:
`static`

Template:
`describe(Pack)`
Mode and number of proofs:
`describe(+atom)` - `zero_or_one`

Exceptions:
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

---

pack_metadata/4

Enumerates by backtracking resolved metadata for installed packs.

Compilation flags:
`static`

Template:
`pack_metadata(Registry,Pack,Version,Metadata)`
Mode and number of proofs:
`pack_metadata(?atom,?atom,?compound,?compound)` - `zero_or_more`

Exceptions:
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`
`Version` is neither a variable nor a compound term:
`type_error(compound,Version)`
`Metadata` is neither a variable nor a compound term:
`type_error(compound,Metadata)`

Remarks:

- `Metadata`: Resolved metadata term for the selected installed pack version.
- Resolved fields: The metadata term includes both version-independent pack metadata and version-specific metadata selected from the installed version.
- Absent optional metadata: Missing optional metadata is normalized to `none` instead of causing failure.
- Metadata term: Metadata is returned using the term `metadata(Name, Description, License, Home, SourceURL, Checksum, Dependencies, Portability, Directory, Pinned, Installed, Loaded)`.

See also:
pack_property/4
pack_object/3
loaded_pack/3

---

pack_property/4

Enumerates by backtracking resolved properties for installed pack versions.

Compilation flags:
`static`

Template:
`pack_property(Registry,Pack,Version,Property)`
Mode and number of proofs:
`pack_property(?atom,?atom,?compound,?compound)` - `zero_or_more`

Exceptions:

`Registry` is neither a variable nor an atom:

`type_error(atom,Registry)`

`Pack` is neither a variable nor an atom:

`type_error(atom,Pack)`

`Version` is neither a variable nor a compound term:

`type_error(compound,Version)`

`Property` is neither a variable nor a compound term:

`type_error(compound,Property)`

Remarks:

- `Property`: A resolved property term for the selected installed pack version.
- Supported properties: Supported properties are `name(Name)`, `description(Description)`, `license(License)`, `home(Home)`, `source_url(URL)`, `checksum(Checksum)`, `dependencies(Dependencies)`, `portability(Portability)`, `directory(Directory)`, `pinned(Boolean)`, `installed(Boolean)`, and `loaded(Boolean)`.
- Absent optional metadata: Missing optional metadata is normalized to `none`.

See also:

pack_metadata/4
pack_object/3

---

pack_object/3

Enumerates by backtracking pack specification objects for defined registry and pack pairs.

Compilation flags:

`static`

Template:

`pack_object(Registry,Pack,PackObject)`

Mode and number of proofs:

`pack_object(?atom,?atom,?atom)` - `zero_or_more`

Exceptions:

`Registry` is neither a variable nor an atom:

`type_error(atom,Registry)`

`Pack` is neither a variable nor an atom:

`type_error(atom,Pack)`

`PackObject` is neither a variable nor an atom:

`type_error(atom,PackObject)`

Remarks:

- `PackObject`: Identifier of the object implementing the `pack_protocol` for the given registry and pack.
- Intended use: This predicate is a low-level bridge for advanced tools and should not be required by most consumers when `pack_metadata/4` or `pack_property/4` is sufficient.

See also:

pack_metadata/4
pack_property/4

---

loaded_pack/3

Enumerates by backtracking all installed packs that contributed loaded files to the current session.

Compilation flags:

`static`

Template:

`loaded_pack(Registry,Pack,Version)`

Mode and number of proofs:

`loaded_pack(?atom,?atom,?compound)` - `zero_or_more`

Exceptions:

`Registry` is neither a variable nor an atom:

`type_error(atom,Registry)`

`Pack` is neither a variable nor an atom:

`type_error(atom,Pack)`

`Version` is neither a variable nor a compound term:

`type_error(compound,Version)`

Remarks:

- Loaded pack: A pack is considered loaded iff at least one currently loaded file belongs to its installation directory.
- `Version`: Installed version of the loaded pack.

See also:

loaded_pack_file/4
pack_metadata/4
installed/3

---

loaded_pack_file/4

Enumerates by backtracking loaded files that belong to installed packs in the current session.

Compilation flags:

`static`

Template:

`loaded_pack_file(Registry,Pack,Version,File)`

Mode and number of proofs:

`loaded_pack_file(?atom,?atom,?compound,?atom)` - `zero_or_more`

Exceptions:

`Registry` is neither a variable nor an atom:

`type_error(atom,Registry)`

`Pack` is neither a variable nor an atom:

`type_error(atom,Pack)`

`Version` is neither a variable nor a compound term:

`type_error(compound,Version)`

`File` is neither a variable nor an atom:

`type_error(atom,File)`

Remarks:

- `File`: Absolute path of a currently loaded file belonging to the selected installed pack.
- Intended use: This predicate supports diagnostics, tests, and the definition of `loaded_pack/3`.

See also:

loaded_pack/3

---

search/1

Searches packs whose name or description includes the search term (case sensitive).

Compilation flags:

`static`

Template:

`search(Term)`

Mode and number of proofs:

`search(+atom)` - `one`

Exceptions:

`Term` is a variable:

`instantiation_error`

`Term` is neither a variable nor an atom:

`type_error(atom,Term)`

---

pack_dependency/6

Enumerates by backtracking resolved direct pack-to-pack dependencies for installed pack versions.

Compilation flags:

`static`

Template:

```
pack_dependency(Registry,Pack,Version,DependencyRegistry,DependencyPack,
DependencyVersion)
```

Mode and number of proofs:

```
pack_dependency(?atom,?atom,?compound,?atom,?atom,?compound) - zero_or_more
```

Exceptions:

`Registry` is neither a variable nor an atom:

```
type_error(atom,Registry)
```

`Pack` is neither a variable nor an atom:

```
type_error(atom,Pack)
```

`Version` is neither a variable nor a compound term:

```
type_error(compound,Version)
```

`DependencyRegistry` is neither a variable nor an atom:

```
type_error(atom,DependencyRegistry)
```

`DependencyPack` is neither a variable nor an atom:

```
type_error(atom,DependencyPack)
```

`DependencyVersion` is neither a variable nor a compound term:

```
type_error(compound,DependencyVersion)
```

Remarks:

- Resolved dependency: Dependencies are resolved against the current installed packs using the same version comparison semantics used by the tool installation logic.
- Supported dependency forms: Version ranges, alternatives, and conjunctions are resolved using the canonical packs dependency semantics.
- Non-pack dependencies: Dependencies on Logtalk, Prolog backends, and operating systems are ignored by this predicate and therefore do not produce results.

See also:

loaded_pack_dependency/6
pack_metadata/4
loaded_pack/3

`loaded_pack_dependency/6`

Enumerates by backtracking resolved direct pack-to-pack dependencies where both the source pack and the dependency pack contributed loaded files to the current session.

Compilation flags:

```
static
```

Template:

```
loaded_pack_dependency(Registry,Pack,Version,DependencyRegistry,DependencyPack,
DependencyVersion)
```

Mode and number of proofs:

```
loaded_pack_dependency(?atom,?atom,?compound,?atom,?atom,?compound) - zero_or_more
```

Exceptions:

`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`
`Version` is neither a variable nor a compound term:
`type_error(compound,Version)`
`DependencyRegistry` is neither a variable nor an atom:
`type_error(atom,DependencyRegistry)`
`DependencyPack` is neither a variable nor an atom:
`type_error(atom,DependencyPack)`
`DependencyVersion` is neither a variable nor a compound term:
`type_error(compound,DependencyVersion)`

Remarks:

- Loaded dependency edge: This predicate is the loaded-pack counterpart of `pack_dependency/6` and only succeeds when both endpoints are currently loaded.
- Intended use: This predicate is especially useful for tools that need to describe relationships between the installed packs that contributed code to the current session.

See also:
pack_dependency/6
loaded_pack/3

`install/4`

Installs a new pack using the specified options. Fails if the pack is unknown or already installed but not using `update(true)` or `force(true)` options. Fails also if the pack version is unknown.

Compilation flags:
`static`

Template:
`install(Registry,Pack,Version,Options)`
Mode and number of proofs:
`install(+atom,+atom,++compound,++list(compound))` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`
`Version` is a variable:
`instantiation_error`

`Version` is neither a variable nor a valid version:
`type_error(pack_version,Version)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

Remarks:

- `update(Boolean)` option: Update pack if already installed. Default is `false`. Overrides the `force/1` option.
- `force(Boolean)` option: Force pack re-installation if already installed. Default is `false`.
- `compatible(Boolean)` option: Restrict installation to compatible packs. Default is `true`.
- `clean(Boolean)` option: Clean pack archive after installation. Default is `false`.
- `verbose(Boolean)` option: Verbose installing steps. Default is `false`.
- `checksum(Boolean)` option: Verify pack archive checksum. Default is `true`.
- `checksig(Boolean)` option: Verify pack archive signature. Default is `false`.
- `git(Atom)` option: Extra command-line options. Default is `''`.
- `downloader(Atom)` option: Downloader utility. Either `curl` or `wget`. Default is `curl`.
- `curl(Atom)` option: Extra command-line options. Default is `''`.
- `wget(Atom)` option: Extra command-line options. Default is `''`.
- `gpg(Atom)` option: Extra command-line options. Default is `''`.
- `tar(Atom)` option: Extra command-line options. Default is `''`.

`install/3`

Installs the specified version of a pack from the given registry using default options. Fails if the pack is already installed or unknown. Fails also if the pack version is unknown.

Compilation flags:
`static`

Template:
`install(Registry,Pack,Version)`
Mode and number of proofs:
`install(+atom,+atom,?compound)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is a variable:

`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`
`Version` is a variable:
`instantiation_error`
`Version` is neither a variable nor a valid version:
`type_error(pack_version,Version)`

See also:
install/4

---

install/2

Installs the latest version of a pack from the given registry using default options. Fails if the pack is already installed or unknown.

Compilation flags:
`static`

Template:
`install(Registry,Pack)`
Mode and number of proofs:
`install(+atom,+atom)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

See also:
install/3

---

install/1

Installs a pack (if its name is unique among all registries) using default options. Fails if the pack is already installed or unknown. Fails also if the pack is available from multiple registries.

Compilation flags:
`static`

Template:

`install(Pack)`

Mode and number of proofs:

`install(+atom)` - `zero_or_one`

Exceptions:

`Pack` is a variable:

`instantiation_error`

`Pack` is not an atom:

`type_error(atom,Pack)`

See also:

install/2

---

update/3

Updates an outdated pack to the specified version using the specified options. Fails if the pack or the pack version is unknown or if the pack is not installed. Fails also if the pack is orphaned or pinned and not using a `force(true)` option.

Compilation flags:

`static`

Template:

`update(Pack,Version,Options)`

Mode and number of proofs:

`update(+atom,++callable,++list(callable))` - `zero_or_one`

Exceptions:

`Pack` is a variable:

`instantiation_error`

`Pack` is neither a variable nor an atom:

`type_error(atom,Pack)`

`Version` is a variable:

`instantiation_error`

`Version` is neither a variable nor a valid version:

`type_error(pack_version,Version)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

Remarks:

- `install(Boolean)` option: Install pack latest version if not already installed. Default is `false`.
- `force(Boolean)` option: Force update if the pack is pinned or breaks installed packs. Default is `false`.
- `compatible(Boolean)` option: Restrict updating to compatible packs. Default is `true`.
- `status(Status)` option: Specify allowed pack status. Default is `[stable,rc,beta,alpha]`. Set to `all` to also allow `experimental` and `deprecated`.
- `clean(Boolean)` option: Clean pack archive after updating. Default is `false`.
- `verbose(Boolean)` option: Verbose updating steps. Default is `false`.
- `checksum(Boolean)` option: Verify pack archive checksum. Default is `true`.
- `checksig(Boolean)` option: Verify pack archive signature. Default is `false`.
- `git(Atom)` option: Extra command-line options. Default is `''`.
- `downloader(Atom)` option: Downloader utility. Either `curl` or `wget`. Default is `curl`.
- `curl(Atom)` option: Extra command-line options. Default is `''`.
- `wget(Atom)` option: Extra command-line options. Default is `''`.
- `gpg(Atom)` option: Extra command-line options. Default is `''`.
- `tar(Atom)` option: Extra command-line options. Default is `''`.

---

update/2

Updates an outdated pack to its latest version using the specified options. Fails if the pack is orphaned, unknown, or not installed. Fails also if the pack is pinned and not using a `force(true)` option.

Compilation flags:

`static`

Template:

`update(Pack,Options)`

Mode and number of proofs:

`update(+atom,++list(callable))` - `zero_or_one`

Exceptions:

`Pack` is a variable:

`instantiation_error`

`Pack` is neither a variable nor an atom:

`type_error(atom,Pack)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

Remarks:

- `install(Boolean)` option: Install pack latest version if not already installed. Default is `false`.

- `force(Boolean)` option: Force update if the pack is pinned or breaks installed packs. Default is `false`.
- `compatible(Boolean)` option: Restrict updating to compatible packs. Default is `true`.
- `status(Status)` option: Specify allowed pack update status. Default is `[stable,rc,beta,alpha]`. Set to `all` to also allow `experimental` and `deprecated`.
- `clean(Boolean)` option: Clean pack archive after updating. Default is `false`.
- `verbose(Boolean)` option: Verbose updating steps. Default is `false`.
- `checksum(Boolean)` option: Verify pack archive checksum. Default is `true`.
- `checksig(Boolean)` option: Verify pack archive signature. Default is `false`.
- `git(Atom)` option: Extra command-line options. Default is `''`.
- `downloader(Atom)` option: Downloader utility. Either `curl` or `wget`. Default is `curl`.
- `curl(Atom)` option: Extra command-line options. Default is `''`.
- `wget(Atom)` option: Extra command-line options. Default is `''`.
- `gpg(Atom)` option: Extra command-line options. Default is `''`.
- `tar(Atom)` option: Extra command-line options. Default is `''`.

---

update/1

Updates an outdated pack to its latest version using default options. Fails if the pack is pinned, orphaned, not installed, unknown, or breaks installed packs.

Compilation flags:

`static`

Template:

`update(Pack)`

Mode and number of proofs:

`update(+atom)` - `zero_or_one`

Exceptions:

`Pack` is a variable:

`instantiation_error`

`Pack` is neither a variable nor an atom:

`type_error(atom,Pack)`

See also:

update/2
update/3

---

update/0

Updates all outdated packs (that are not pinned) using default options.

Compilation flags:

`static`

Mode and number of proofs:

`update` - `zero_or_one`

`uninstall/2`

Uninstalls a pack using the specified options. Fails if the pack is unknown or not installed. Fails also if the pack is pinned or have dependents and not using a `force(true)` option.

Compilation flags:
`static`

Template:
`uninstall(Pack,Options)`
Mode and number of proofs:
`uninstall(+atom,++list(compound))` - `zero_or_one`

Exceptions:
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

Remarks:

- `force(Boolean)` option: Force deletion if the pack is pinned. Default is `false`.
- `clean(Boolean)` option: Clean pack archive after deleting. Default is `false`.
- `verbose(Boolean)` option: Verbose uninstalling steps. Default is `false`.

`uninstall/1`

Uninstalls a pack using default options. Fails if the pack is pinned, have dependents, not installed, or unknown.

Compilation flags:
`static`

Template:
`uninstall(Pack)`

Mode and number of proofs:
`uninstall(+atom)` - `zero_or_one`

Exceptions:
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

See also:
uninstall/2

---

uninstall/0

Uninstalls all packs using the `force(true)` option.

Compilation flags:
`static`

Mode and number of proofs:
`uninstall` - `zero_or_one`

---

clean/2

Cleans all pack archives. Fails if the the pack is unknown.

Compilation flags:
`static`

Template:
`clean(Registry,Pack)`
Mode and number of proofs:
`clean(+atom,+atom)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

---

clean/1

Cleans all pack archives. Fails if the pack is unknown.

Compilation flags:
`static`

Template:
`clean(Pack)`
Mode and number of proofs:
`clean(+atom)` - `zero_or_one`

Exceptions:
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

See also:
clean/2

---

clean/0

Cleans all archives for all packs.

Compilation flags:
`static`

Mode and number of proofs:
`clean` - `one`

---

save/2

Saves a list of all installed packs and registries plus pinning status to a file using the given options.

Compilation flags:
`static`

Template:
`save(File,Options)`
Mode and number of proofs:
`save(+atom,++list(compound))` - `one_or_error`

Exceptions:
`File` is a variable:

`instantiation_error`

`File` is neither a variable nor an atom:

`type_error(atom,File)`

`File` is an existing file but cannot be written:

`permission_error(open,source_sink,File)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

Lock mode is requested with checksums disabled:

`consistency_error(compatible_options,lock(true),checksum(false))`

The setup cannot be represented by a complete lock file:

`domain_error(lock_setup,Reason)`

Remarks:

- `lock(Boolean)` option: Save complete version 1 lock facts for git registries and SHA-256 archive packs. Requires `checksum(true)` and a satisfiable locked dependency graph. Default is `false`.
- `save(What)` option: Save registries without installed packs when `What` is `all` and skip them when `What` is `installed`. Default is `installed`.

`save/1`

Saves a list of all installed packs and their registries plus pinning status to a file using default options.

Compilation flags:

`static`

Template:

`save(File)`

Mode and number of proofs:

`save(+atom)` - `one_or_error`

Exceptions:

`File` is a variable:

`instantiation_error`

`File` is neither a variable nor an atom:

`type_error(atom,File)`

`File` is an existing file but cannot be written:

`permission_error(open,source_sink,File)`

See also:

save/2

restore/2

Restores a list of registries and packs plus their pinning status from a file using the given options. Fails if restoring is not possible.

Compilation flags:

`static`

Template:

`restore(File,Options)`

Mode and number of proofs:

`restore(+atom,++list(compound))` - `zero_or_one_or_error`

Exceptions:

`File` is a variable:

`instantiation_error`

`File` is neither a variable nor an atom:

`type_error(atom,File)`

`File` is an atom but not an existing file:

`existence_error(file,File)`

`File` is an existing file but cannot be read:

`permission_error(open,source_sink,File)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

Lock mode is requested with checksums disabled:

`consistency_error(compatible_options,lock(true),checksum(false))`

Remarks:

- `lock(Boolean)` option: Require complete version 1 lock facts, git registries, SHA-256 archive packs, and exact dependency versions. Requires `checksum(true)`. Restoring or verification failure is fatal; completed changes are not rolled back. Default is `false`.
- `force(Boolean)` option: Force restoring if a registry is already defined or a pack is already installed. Default is `true`.
- `compatible(Boolean)` option: Restrict installation to compatible packs. Default is `true`.
- `clean(Boolean)` option: Clean registry and pack archives after restoring. Default is `false`.
- `verbose(Boolean)` option: Verbose restoring steps. Default is `false`.
- `checksum(Boolean)` option: Verify pack archive checksums. Must be `true` in lock mode. Default is `true`.
- `checksig(Boolean)` option: Verify pack archive signatures. Default is `false`.
- `git(Atom)` option: Extra command-line options. Default is `''`.
- `downloader(Atom)` option: Downloader utility. Either `curl` or `wget`. Default is `curl`.

- `curl(Atom)` option: Extra command-line options. Default is `''`.
- `wget(Atom)` option: Extra command-line options. Default is `''`.
- `gpg(Atom)` option: Extra command-line options. Default is `''`.
- `tar(Atom)` option: Extra command-line options. Default is `''`.

`restore/1`

Restores a list of registries and packs plus their pinning status from a file using default options. Fails if restoring is not possible.

Compilation flags:
`static`

Template:
`restore(File)`
Mode and number of proofs:
`restore(+atom)` - `zero_or_one_or_error`

Exceptions:
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`File` is an atom but not an existing file:
`existence_error(file,File)`
`File` is an existing file but cannot be read:
`permission_error(open,source_sink,File)`

See also:
restore/2

`dependents/3`

Returns a list of all installed packs that depend on the given pack from the given registry. Fails if the pack is unknown.

Compilation flags:
`static`

Template:
`dependents(Registry,Pack,Dependents)`
Mode and number of proofs:
`dependents(+atom,+atom,-list(atom))` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:

`type_error(atom,Registry)`
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

---

dependents/2

Prints a list of all installed packs that depend on the given pack from the given registry. Fails if the pack is unknown.

Compilation flags:
`static`

Template:
`dependents(Registry,Pack)`
Mode and number of proofs:
`dependents(+atom,+atom)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

---

dependents/1

Prints a list of all installed packs that depend on the given pack if unique from all defined registries. Fails if the pack is unknown or available from multiple registries.

Compilation flags:
`static`

Template:
`dependents(Pack)`
Mode and number of proofs:
`dependents(+atom)` - `zero_or_one`

Exceptions:
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:

```
type_error(atom,Pack)
```

---

lint/2

Checks the pack specification. Fails if the pack is unknown or if linting detects errors.

Compilation flags:
`static`

Template:
`lint(Registry,Pack)`
Mode and number of proofs:
`lint(+atom,+atom)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

---

lint/1

Checks the pack specification. Fails if the pack is unknown, or available from multiple registries, or if linting detects errors.

Compilation flags:
`static`

Template:
`lint(Pack)`
Mode and number of proofs:
`lint(+atom)` - `zero_or_one`

Exceptions:
`Pack` is a variable:
`instantiation_error`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

---

lint/0

Checks all pack specifications.

Compilation flags:
`static`

Mode and number of proofs:
`lint` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

save_setup_terms/2

Collects the setup facts before opening the destination file.

Compilation flags:
`static`

Template:
`save_setup_terms(Options,Terms)`
Mode and number of proofs:
`save_setup_terms(+list(compound),-list(compound))` - `one_or_error`

Exceptions:
The setup cannot be represented by a complete lock file:
`domain_error(lock_setup,Reason)`

---

lock_registry_facts/3

Collects the URLs and full commits of all selected clean git registries.

Compilation flags:
`static`

Template:
`lock_registry_facts(Registries,Terms,Commits)`
Mode and number of proofs:
`lock_registry_facts(+list(atom),-list(compound),-list(compound))` - `one_or_error`

Exceptions:
A selected registry has no usable git commit or has tracked changes:

```
domain_error(lock_setup,registry(Registry))
```

---

lock_pack_facts/2

Collects the declared SHA-256 archive identities of all selected packs.

Compilation flags:
`static`

Template:
`lock_pack_facts(Packs,Terms)`
Mode and number of proofs:
`lock_pack_facts(+list(compound),-list(compound))` - `one_or_error`

Exceptions:
A selected pack has no usable SHA-256 archive identity:
`domain_error(lock_setup,pack(Registry,Pack,Version))`

---

check_lock_options/1

Checks that lock mode requires archive checksum verification.

Compilation flags:
`static`

Template:
`check_lock_options(Options)`
Mode and number of proofs:
`check_lock_options(+list(compound))` - `one_or_error`

Exceptions:
Lock mode is requested with checksums disabled:
`consistency_error(compatible_options,lock(true),checksum(false))`

Operators

(none)
category

1.249.3 packs_common

Common predicates for the packs tool objects.

Availability:
`logtalk_load(packs(loader))`

Author: Paulo Moura
Version: 0:33:0
Date: 2025-01-23

Compilation flags:
`static`

Provides:
type::type/1
type::check/2
Uses:
list
logtalk
os
type
user

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *setup/0*
  - *reset/0*
  - *verify_commands_availability/0*
  - *help/0*
  - *pin/1*
  - *pin/0*
  - *unpin/1*
  - *unpin/0*
  - *pinned/1*
  - *directory/2*
  - *directory/1*
  - *readme/2*
  - *readme/1*
  - *logtalk_packs/1*
  - *logtalk_packs/0*
  - *prefix/1*

    - *prefix/0*
- *Protected predicates*
    - *readme_file_path/2*
    - *print_readme_file_path/1*
    - *command/2*
    - *load_registry/1*
    - *sha256sum_command/1*
    - *tar_command/1*
    - *supported_archive/1*
    - *supported_url_archive/1*
    - *decode_url_spaces/2*
- *Private predicates*
- *Operators*

Public predicates

setup/0

Ensures that registries and packs directory structure exists. Preserves any defined registries and installed packs.

Compilation flags:

`static`

Mode and number of proofs:

`setup` - `one`

---

reset/0

Resets registries and packs directory structure. Deletes any defined registries and installed packs.

Compilation flags:

`static`

Mode and number of proofs:

`reset` - `one`

---

verify_commands_availability/0

Verifies required shell commands availability. Fails printing an error message if a command is missing.

Compilation flags:
`static`

Mode and number of proofs:
`verify_commands_availability` - `zero_or_one`

---

help/0

Provides help about the main predicates.

Compilation flags:
`static`

Mode and number of proofs:
`help` - `one`

---

pin/1

Pins a resource (pack or registry) preventing it from being updated, uninstalled, or deleted. Fails if the resource is not found.

Compilation flags:
`static`

Template:
`pin(Resource)`
Mode and number of proofs:
`pin(+atom)` - `zero_or_one`

Exceptions:
`Resource` is a variable:
`instantiation_error`
`Resource` is neither a variable nor an atom:
`type_error(atom,Resource)`

---

pin/0

Pins all resource (packs or registries) preventing them from being updated, uninstalled, or deleted. Note that resources added after calling this predicate will not be pinned.

Compilation flags:
`static`

Mode and number of proofs:
`pin` - `one`

---

unpin/1

Unpins a resource (pack or registry), allowing it to be updated, uninstalled, or deleted. Fails if the resource is not found.

Compilation flags:
`static`

Template:
`unpin(Resource)`
Mode and number of proofs:
`unpin(+atom)` - `zero_or_one`

Exceptions:
`Resource` is a variable:
`instantiation_error`
`Resource` is neither a variable nor an atom:
`type_error(atom,Resource)`

---

unpin/0

Unpins all resources (packs or registries), allowing them to be updated, uninstalled, or deleted.

Compilation flags:
`static`

Mode and number of proofs:
`unpin` - `one`

---

pinned/1

True iff the resource (pack or registry) is defined or installed and if it is pinned.

Compilation flags:
`static`

Template:
`pinned(Resource)`
Mode and number of proofs:
`pinned(+atom)` - `zero_or_one`

Exceptions:
`Resource` is a variable:
`instantiation_error`
`Resource` is neither a variable nor an atom:
`type_error(atom,Resource)`

---

directory/2

Enumerates by backtracking all packs or registries and respective installation or definition directories (using the internal backend format).

Compilation flags:
`static`

Template:
`directory(Resource,Directory)`
Mode and number of proofs:
`directory(?atom,?atom)` - `zero_or_more`

Exceptions:
`Resource` is neither a variable nor an atom:
`type_error(atom,Resource)`
`Directory` is neither a variable nor an atom:
`type_error(atom,Directory)`

---

directory/1

Prints the directory where the registry or the pack is installed (using the native operating-system format).

Compilation flags:
`static`

Template:

```
directory(Resource)
```

Mode and number of proofs:

```
directory(+atom) - zero_or_one
```

Exceptions:

`Resource` is a variable:

```
instantiation_error
```

`Resource` is neither a variable nor an atom:

```
type_error(atom,Resource)
```

---

readme/2

Returns the path to the resource (pack or registry) readme file (using the internal backend format). Fails if the resource is not defined or installed or if no readme file is found for it.

Compilation flags:

```
static
```

Template:

```
readme(Resource,ReadMeFile)
```

Mode and number of proofs:

```
readme(+atom,-atom) - zero_or_one
```

Exceptions:

`Resource` is a variable:

```
instantiation_error
```

`Resource` is neither a variable nor an atom:

```
type_error(atom,Resource)
```

`ReadMeFile` is neither a variable nor an atom:

```
type_error(atom,ReadMeFile)
```

---

readme/1

Prints the path to the resource (pack or registry) readme file (using the native operating-system format). Fails if the resource is not defined or installed or if no readme file is found for it.

Compilation flags:

```
static
```

Template:

```
readme(Resource)
```

Mode and number of proofs:

```
readme(+atom) - zero_or_one
```

Exceptions:

`Resource` is a variable:
`instantiation_error`
`Resource` is neither a variable nor an atom:
`type_error(atom,Resource)`

`logtalk_packs/1`

Returns the directory prefix (using the internal backend format) where the registries, packs, and archives are installed.

Compilation flags:
`static`

Template:
`logtalk_packs(LogtalkPacks)`
Mode and number of proofs:
`logtalk_packs(-atom)` - `one`

Exceptions:
`LogtalkPacks` is neither a variable nor an atom:
`type_error(atom,LogtalkPacks)`

`logtalk_packs/0`

Prints the directory prefix (using the native operating-system format) where the registries, packs, and archives are installed.

Compilation flags:
`static`

Mode and number of proofs:
`logtalk_packs` - `one`

`prefix/1`

Returns the directory prefix (using the internal backend format) where the registries or packs are installed.

Compilation flags:
`static`

Template:
`prefix(Prefix)`
Mode and number of proofs:
`prefix(-atom)` - `one`

Exceptions:

`Prefix` is neither a variable nor an atom:

`type_error(atom,Prefix)`

`prefix/0`

Prints the directory prefix (using the native operating-system format) where the registries or packs are installed.

Compilation flags:

`static`

Mode and number of proofs:

`prefix` - `one`

Protected predicates

`readme_file_path/2`

Returns the absolute path for the given directory readme file if it exists.

Compilation flags:

`static`

Template:

`readme_file_path(Directory,ReadMeFile)`

Mode and number of proofs:

`readme_file_path(+atom,-atom)` - `zero_or_one`

Remarks:

- Valid file names: Case variations of `README` and `NOTES` with or without a `.md` or `.txt` extension. The recommended file name is `README.md`.

`print_readme_file_path/1`

Prints the absolute path for the given directory readme file if it exists. Succeeds otherwise.

Compilation flags:

`static`

Template:

`print_readme_file_path(Directory)`

Mode and number of proofs:

`print_readme_file_path(+atom)` - `one`

---

command/2

Executes a shell command. Prints an error message and fails if the command fails.

Compilation flags:

```
static
```

Template:

```
command(Command,FailureMessage)
```

Mode and number of proofs:

```
command(+atom,@nonvar) - zero_or_one
```

---

load_registry/1

Loads all registry files from the given directory.

Compilation flags:

```
static
```

Template:

```
load_registry(Directory)
```

Mode and number of proofs:

```
load_registry(+atom) - zero_or_one
```

---

sha256sum_command/1

Returns the name of the sha256sum command to be used on POSIX systems. Fails if neither gsha256sum or sha256sum commands are found.

Compilation flags:

```
static
```

Template:

```
sha256sum_command(Command)
```

Mode and number of proofs:

```
sha256sum_command(-atom) - zero_or_one
```

---

tar_command/1

Returns the name of the tar command to be used depending on the operating-system.

Compilation flags:
`static`

Template:
`tar_command(Command)`
Mode and number of proofs:
`tar_command(-atom)` - `one`

---

supported_archive/1

True iff the archive format is supported.

Compilation flags:
`static`

Template:
`supported_archive(Extension)`
Mode and number of proofs:
`supported_archive(+atom)` - `zero_or_one`

---

supported_url_archive/1

True iff the URL archive is supported.

Compilation flags:
`static`

Template:
`supported_url_archive(URL)`
Mode and number of proofs:
`supported_url_archive(+atom)` - `zero_or_one`

`decode_url_spaces/2`

Decodes encoded spaces (%20) in URLs to spaces.

Compilation flags:
`static`

Template:
`decode_url_spaces(URL,Decoded)`
Mode and number of proofs:
`decode_url_spaces(+atom,-atom) - one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.249.4 `packs_messages`

Packs default message translations.

Availability:
`logtalk_load(packs(loader))`

Author: Paulo Moura
Version: 0:43:0
Date: 2026-10-08

Compilation flags:
`static`

Provides:
logtalk::message_prefix_stream/4
logtalk::message_tokens//2

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.249.5 `packs_specs_hook`

Hook object for filtering registry and pack specification file contents.

Availability:
`logtalk_load(packs(loader))`

Author: Paulo Moura
Version: 0:13:0
Date: 2022-06-28

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
character
logtalk

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.249.6 `registries`

Registry handling predicates.

Availability:
`logtalk_load(packs(loader))`

Author: Paulo Moura
Version: 0:65:1
Date: 2026-10-31

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *packs_common*
`public` *options*
Uses:
list
logtalk
os
type
user

Remarks:
(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *directory/1* *directory/2* *help/0* *logtalk_packs/0* *logtalk_packs/1* *option/2* *option/3* *pin/0* *pin/1* *pinned/1* *prefix/0* *prefix/1* *readme/1* *readme/2* *reset/0* *setup/0* *unpin/0* *unpin/1* *valid_option/1* *valid_options/1* *verify_commands_availability/0*

- *Public predicates*
  - *list/0*
  - *describe/1*
  - *defined/4*
  - *add/3*
  - *add/2*
  - *add/1*
  - *update/2*
  - *update/1*
  - *update/0*
  - *delete/2*
  - *delete/1*
  - *delete/0*
  - *clean/1*
  - *clean/0*
  - *provides/2*
  - *lint/1*
  - *lint/0*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

list/0

Prints a list of all defined registries, including how defined (`git`, `archive`, or `directory`) and if they are pinned.

Compilation flags:

`static`

Mode and number of proofs:

`list` - `one`

describe/1

Prints all registry entries.

Compilation flags:
`static`

Template:
`describe(Registry)`
Mode and number of proofs:
`describe(+atom)` - `one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`

---

defined/4

Enumerates by backtracking all defined registries, their definition URL, how they are defined (`git`, `archive`, or `directory`), and if they are pinned.

Compilation flags:
`static`

Template:
`defined(Registry,URL,HowDefined,Pinned)`
Mode and number of proofs:
`defined(?atom,?atom,?atom,?boolean)` - `zero_or_more`

Exceptions:
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`URL` is neither a variable nor an atom:
`type_error(atom,URL)`
`HowDefined` is neither a variable nor an atom:
`type_error(atom,HowDefined)`
`Pinned` is neither a variable nor a boolean:
`type_error(boolean,Pinned)`

---

add/3

Adds a new registry using the given options. Fails if the registry cannot be added or if it is already defined but not using `update(true)` or `force(true)` options. A `file://` URL can be used for a local directory or archive.

Compilation flags:

`static`

Template:

`add(Registry,URL,Options)`

Mode and number of proofs:

`add(+atom,+atom,++list(compound))` - `zero_or_one`

Exceptions:

`Registry` is a variable:

`instantiation_error`

`Registry` is neither a variable nor an atom:

`type_error(atom,Registry)`

`URL` is a variable:

`instantiation_error`

`URL` is neither a variable nor an atom:

`type_error(atom,URL)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

Remarks:

- Registry name: Must be the URL basename when using a git URL or a local directory URL. Must also be the declared registry name in the registry specification object.
- HTTPS URLs: Must end with either a `.git` extension or an archive extension.
- `commit(Hash)` option: When present, checkout the registry git repository at the specified commit hash after cloning.
- `update(Boolean)` option: Update registry if already defined. Default is `false`. Overrides the `force/1` option.
- `force(Boolean)` option: Force registry re-installation if already defined by first deleting the previous installation. Default is `false`.
- `clean(Boolean)` option: Clean registry archive after updating. Default is `false`.
- `verbose(Boolean)` option: Verbose adding steps. Default is `false`.
- `downloader(Atom)` option: Downloader utility. Either `curl` or `wget`. Default is `curl`.
- `curl(Atom)` option: Extra command-line options. Default is `''`.
- `wget(Atom)` option: Extra command-line options. Default is `''`.
- `gpg(Atom)` option: Extra command-line options. Default is `''`.
- `tar(Atom)` option: Extra command-line options. Default is `''`.

add/2

Adds a new registry using default options. Fails if the registry cannot be added or if it is already defined. HTTPS URLs must end with either a `.git` extension or an archive extension. A `file://` URL can be used for a local directory or archive.

Compilation flags:
`static`

Template:
`add(Registry,URL)`
Mode and number of proofs:
`add(+atom,+atom)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`URL` is a variable:
`instantiation_error`
`URL` is neither a variable nor an atom:
`type_error(atom,URL)`

Remarks:

- Registry name: Must be the URL basename when using a git URL or a local directory URL. Must also be the declared registry name in the registry specification object.

See also:
add/3

---

add/1

Adds a new registry using default options. Fails if the registry cannot be added or if it is already defined. HTTPS URLs must end with a `.git` extension or an archive extension. A `file://` URL can be used for a local directory or archive.

Compilation flags:
`static`

Template:
`add(URL)`
Mode and number of proofs:
`add(+atom)` - `zero_or_one`

Exceptions:
`URL` is a variable:
`instantiation_error`
`URL` is neither a variable nor an atom:

`type_error(atom,URL)`

Remarks:

- Registry name: Taken from the URL basename.

See also:
add/2

---

update/2

Updates a defined registry using the specified options. Fails if the registry is not defined.

Compilation flags:
`static`

Template:
`update(Registry,Options)`
Mode and number of proofs:
`update(+atom,++list(compound))` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

Remarks:

- `commit(Hash)` option: When present, checkout the registry git repository at the specified commit hash after updating.
- `force(Boolean)` option: Force update if the registry is pinned. Default is `false`.
- `clean(Boolean)` option: Clean registry archive after updating. Default is `false`.
- `verbose(Boolean)` option: Verbose updating steps. Default is `false`.
- `downloader(Atom)` option: Downloader utility. Either `curl` or `wget`. Default is `curl`.
- `curl(Atom)` option: Extra command-line options. Default is `''`.
- `wget(Atom)` option: Extra command-line options. Default is `''`.
- `gpg(Atom)` option: Extra command-line options. Default is `''`.
- `tar(Atom)` option: Extra command-line options. Default is `''`.

---

update/1

Updates a defined registry using default options. Fails if the registry is not defined.

Compilation flags:
`static`

Template:
`update(Registry)`
Mode and number of proofs:
`update(+atom)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`

See also:
update/2

---

update/0

Updates all defined registries using default options.

Compilation flags:
`static`

Mode and number of proofs:
`update` - `zero_or_one`

---

delete/2

Deletes a registry using the specified options (if not pinned).

Compilation flags:
`static`

Template:
`delete(Registry,Options)`
Mode and number of proofs:
`delete(+atom,++list(compound))` - `zero_or_one`

Exceptions:
`Registry` is a variable:

`instantiation_error`

`Registry` is neither a variable nor an atom:

`type_error(atom,Registry)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

Remarks:

- `force(Boolean)` option: Force deletion if the registry is pinned or there are installed registry packs. Default is `false`.
- `clean(Boolean)` option: Clean registry archive after deleting. Default is `false`.
- `verbose(Boolean)` option: Verbose deleting steps. Default is `false`.
- `downloader(Atom)` option: Downloader utility. Either `curl` or `wget`. Default is `curl`.
- `curl(Atom)` option: Extra command-line options. Default is `''`.
- `wget(Atom)` option: Extra command-line options. Default is `''`.
- `gpg(Atom)` option: Extra command-line options. Default is `''`.
- `tar(Atom)` option: Extra command-line options. Default is `''`.

`delete/1`

Deletes a registry using default options.

Compilation flags:

`static`

Template:

`delete(Registry)`

Mode and number of proofs:

`delete(+atom)` - `zero_or_one`

Exceptions:

`Registry` is a variable:

`instantiation_error`

`Registry` is neither a variable nor an atom:

`type_error(atom,Registry)`

See also:

delete/2

---

delete/0

Deletes all registries using the `force(true)` option.

Compilation flags:
`static`

Mode and number of proofs:
`delete` - `zero_or_one`

---

clean/1

Cleans all registry archives. Fails if the registry is not defined.

Compilation flags:
`static`

Template:
`clean(Registry)`
Mode and number of proofs:
`clean(+atom)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`

---

clean/0

Cleans all archives for all registries.

Compilation flags:
`static`

Mode and number of proofs:
`clean` - `one`

---

provides/2

Enumerates by backtracking all packs provided by a registry.

Compilation flags:
`static`

Template:
`provides(Registry,Pack)`
Mode and number of proofs:
`provides(?atom,?atom)` - `zero_or_more`

Exceptions:
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`
`Pack` is neither a variable nor an atom:
`type_error(atom,Pack)`

---

lint/1

Checks the registry specification. Fails if the registry is not defined or if linting detects errors.

Compilation flags:
`static`

Template:
`lint(Registry)`
Mode and number of proofs:
`lint(+atom)` - `zero_or_one`

Exceptions:
`Registry` is a variable:
`instantiation_error`
`Registry` is neither a variable nor an atom:
`type_error(atom,Registry)`

---

lint/0

Checks all registry specifications.

Compilation flags:
`static`

Mode and number of proofs:
`lint` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.249.7 `registry_loader_hook`

Hook object for filtering registry loader file contents.

Availability:
`logtalk_load(packs(loader))`

Author: Paulo Moura
Version: 0:13:0
Date: 2022-11-20

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
character
logtalk

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.249.8 `registry_protocol`

Registry specification protocol. Objects implementing this protocol should be named after the pack with a `_registry` suffix and saved in a file with the same name as the object.

Availability:
`logtalk_load(packs(loader))`

Author: Paulo Moura
Version: 0:12:0
Date: 2022-06-28

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *name/1*
  - *description/1*
  - *home/1*
  - *clone/1*
  - *archive/1*
  - *note/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

name/1

Registry name. Preferably a valid unquoted atom.

Compilation flags:
`static`

Template:
`name(Name)`
Mode and number of proofs:
`name(?atom)` - `zero_or_one`

---

description/1

Registry one line description.

Compilation flags:
`static`

Template:
`description(Description)`
Mode and number of proofs:
`description(?atom)` - `zero_or_one`

---

home/1

Registry home HTTPS or file URL.

Compilation flags:
`static`

Template:
`home(Home)`
Mode and number of proofs:
`home(?atom)` - `zero_or_one`

---

clone/1

Registry git clone HTTPS URL (must end with the `.git` extension). Git repos should have the same name as the registry.

Compilation flags:
`static`

Template:
`clone(URL)`
Mode and number of proofs:
`clone(?atom)` - `zero_or_one`

---

archive/1

Registry archive download HTTPS URL.

Compilation flags:
`static`

Template:
`archive(URL)`
Mode and number of proofs:
`archive(?atom)` - `zero_or_one`

---

note/2

Table of notes per action.

Compilation flags:
`static`

Template:
`note(Action,Note)`
Mode and number of proofs:
`note(?atom,-atom)` - `zero_or_more`

Remarks:

- Action: Possible values are `add`, `update`, and `delete`. When unbound, the note apply to all actions.
- Note: Note to print when performing an action on a registry.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.250 particle_swarm_optimization

object

1.250.1 `particle_swarm_optimization(Problem)`

- `Problem` - Problem object implementing `particle_swarm_optimization_protocol`.

Particle swarm optimization algorithm using the Xoshiro128++ random number generator via `fast_random(xoshiro128pp)`.

Availability:
`logtalk_load(particle_swarm_optimization(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *particle_swarm_optimization(Problem,fast_random(xoshiro128pp))*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

particle_swarm_optimization(Problem,Random), *particle_swarm_optimization_protocol*

object

1.250.2 `particle_swarm_optimization(Problem,Random)`

- `Problem` - Problem object implementing `particle_swarm_optimization_protocol`.
- `Random` - Random number generator object providing `between/3`, `random/1`, and `randomize/1` predicates.

Continuous bounded global-best particle swarm optimization algorithm. Parameterized by a problem object implementing the `particle_swarm_optimization_protocol` protocol and by a random number generator algorithm for the `fast_random` library. The algorithm minimizes or maximizes the fitness function defined by the problem.

Availability:

```
logtalk_load(particle_swarm_optimization(loader))
```

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *options*

Uses:

linear_algebra
numberlist
type

Remarks:

- Algorithm: Uses synchronous global-best particle swarm optimization. Every particle update in an iteration uses the global best from the start of that iteration.
- Optimization objective: The `objective(minimize|maximize)` option selects the fitness ordering. Fitness values are otherwise used unchanged.

- Target fitness: The `target_fitness(Fitness)` option stops the run when the best fitness reaches or passes the target in the selected objective direction.
- Stagnation stopping: The `stagnation_iterations(N)` option stops the run after `N` consecutive iterations without a strict global-best improvement; zero disables this condition.
- Initial velocities: If the problem defines `initial_velocities/1`, its velocities are validated and used. Otherwise, velocities are sampled randomly.
- Boundary handling: Velocities are limited to plus or minus the range of each dimension. A position crossing a bound is clamped to that bound and its velocity component is set to zero.
- Progress reporting: If the problem object defines `progress/5`, it is called periodically and once when the loop terminates.
- Seed control: The `seed(S)` option initializes the random number generator for reproducible runs.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *run/2*
  - *run/3*
  - *run/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

run/2

Runs the particle swarm optimization algorithm using default options and returns the best position and fitness found.

Compilation flags:

`static`

Template:

`run(BestPosition,BestFitness)`

Mode and number of proofs:

`run(-list(number),-number)` - `one`

---

run/3

Runs the particle swarm optimization algorithm using the given options and returns the best position and fitness found.

Compilation flags:

`static`

Template:

`run(BestPosition,BestFitness,Options)`

Mode and number of proofs:

`run(-list(number),-number,+list(compound))` - `one`

Remarks:

- `objective(Objective)` option: Optimization objective, either `minimize` or `maximize` (default: `minimize`).
- `target_fitness(Fitness)` option: Numeric target that stops the run when reached or passed in the selected objective direction (default: `none`).
- `max_iterations(N)` option: Maximum number of swarm iterations (default: `1000`).
- `stagnation_iterations(N)` option: Number of consecutive iterations without a strict global-best improvement before stopping; zero disables this condition (default: `0`).
- `inertia_weight(W)` option: Velocity inertia weight (default: `0.7298`).
- `cognitive_coefficient(C)` option: Personal-best acceleration coefficient (default: `1.49618`).
- `social_coefficient(C)` option: Global-best acceleration coefficient (default: `1.49618`).
- `updates(N)` option: Number of progress reports during the run; zero disables reporting (default: `0`).
- `seed(S)` option: Positive integer random seed for reproducible runs.

`run/4`

Runs the particle swarm optimization algorithm using the given options and returns the best position, best fitness, and run statistics.

Compilation flags:

`static`

Template:

`run(BestPosition,BestFitness,Statistics,Options)`

Mode and number of proofs:

`run(-list(number),-number,-list(compound),+list(compound))` - `one`

Remarks:

- Statistics list: A list containing `iterations(N)`, `evaluations(E)`, `improvements(I)`, `final_mean_fitness(M)`, and `final_diversity(D)`. Improvements are measured in the selected objective direction.

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

particle_swarm_optimization(Problem), *particle_swarm_optimization_protocol*

protocol

1.250.3 particle_swarm_optimization_protocol

Protocol for continuous bounded particle swarm optimization problem definitions. A problem object must define the three required predicates and may optionally define predicates for initial velocities, stopping, and progress reporting.

Availability:
`logtalk_load(particle_swarm_optimization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *initial_positions/1*
  - *position_bounds/1*
  - *initial_velocities/1*
  - *fitness/2*
  - *stop_condition/3*
  - *progress/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

initial_positions/1

Returns a non-empty list of initial particle positions. All positions must be non-empty lists of numbers with the same dimensions as the position bounds.

Compilation flags:
`static`

Template:
`initial_positions(Positions)`
Mode and number of proofs:
`initial_positions(-list(list(number)))` - `one`

---

position_bounds/1

Returns one `Lower-Upper` numeric bound pair per position dimension, with `Lower =< Upper`.

Compilation flags:
`static`

Template:
`position_bounds(Bounds)`
Mode and number of proofs:
`position_bounds(-list(pair))` - `one`

---

initial_velocities/1

Returns one initial velocity vector per initial position. Each vector must have the same dimensions as the position bounds, with every component between plus or minus the corresponding position range. Optional. When not defined or when it fails, velocities are initialized randomly.

Compilation flags:
`static`

Template:
`initial_velocities(Velocities)`
Mode and number of proofs:
`initial_velocities(-list(list(number)))` - `zero_or_one`

---

fitness/2

Computes the fitness of a particle position. The optimization direction is selected using the optimizer `objective/1` option.

Compilation flags:

```
static
```

Template:

```
fitness(Position,Fitness)
```

Mode and number of proofs:

```
fitness(+list(number),-number) - one
```

---

stop_condition/3

True when the search should stop given the completed iteration count, best position, and best fitness. Optional. When not defined, the search runs for the configured maximum number of iterations.

Compilation flags:

```
static
```

Template:

```
stop_condition(Iteration,BestPosition,BestFitness)
```

Mode and number of proofs:

```
stop_condition(+non_negative_integer,+list(number),+number) - zero_or_one
```

---

progress/5

Called periodically to report the completed iteration count, best position, best fitness, mean swarm fitness, and swarm diversity. Diversity is the mean Euclidean distance from the swarm centroid. Optional.

Compilation flags:

```
static
```

Template:

```
progress(Iteration,BestPosition,BestFitness,MeanFitness,Diversity)
```

Mode and number of proofs:

```
progress(+non_negative_integer,+list(number),+number,+number,+number) -
zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

particle_swarm_optimization(Problem)

1.251 partitions

object

1.251.1 `partitions`

Implementation of set partition operations over lists.

Availability:
`logtalk_load(partitions(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *partitions_protocol*
Uses:
fast_random(Algorithm)
integer
list
natural

Remarks:
(none)

Inherited public predicates:
count_distinct_partitions/2 *count_distinct_partitions/3* *count_partitions/2* *count_partitions/3* *distinct_partition/2* *distinct_partition/3* *distinct_partition/4* *distinct_partition_index/3* *distinct_partition_index/4* *distinct_partition_index/5* *distinct_partitions/2* *distinct_partitions/3* *distinct_partitions/4* *next_partition/3* *next_partition/4* *nth_distinct_partition/3* *nth_distinct_partition/4* *nth_distinct_partition/5* *nth_partition/3* *nth_partition/4* *nth_partition/5* *partition/2* *partition/3* *partition/4* *partition_index/3* *partition_index/4* *partition_index/5* *partitions/2* *partitions/3* *partitions/4* *previous_partition/3*

previous_partition/4 *random_distinct_partition/2* *random_distinct_partition/3* *random_partition/2* *random_partition/3* *sample_distinct_partitions/3* *sample_distinct_partitions/4* *sample_partitions/3* *sample_partitions/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.251.2 `partitions_protocol`

Protocol for set partition operations over lists.

Availability:
`logtalk_load(partitions(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-12

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *partitions/2*
  - *partition/2*
  - *partitions/3*
  - *partition/3*
  - *partitions/4*
  - *partition/4*
  - *distinct_partitions/2*
  - *distinct_partition/2*
  - *distinct_partitions/3*
  - *distinct_partition/3*
  - *distinct_partitions/4*
  - *distinct_partition/4*
  - *count_partitions/2*
  - *nth_partition/3*
  - *nth_partition/4*
  - *nth_partition/5*
  - *partition_index/3*
  - *partition_index/4*
  - *partition_index/5*
  - *count_partitions/3*
  - *count_distinct_partitions/2*
  - *nth_distinct_partition/3*
  - *nth_distinct_partition/4*
  - *nth_distinct_partition/5*
  - *distinct_partition_index/3*
  - *distinct_partition_index/4*
  - *distinct_partition_index/5*
  - *count_distinct_partitions/3*
  - *random_partition/2*
  - *random_partition/3*
  - *sample_partitions/3*
  - *sample_partitions/4*
  - *random_distinct_partition/2*
  - *random_distinct_partition/3*
  - *sample_distinct_partitions/3*
  - *sample_distinct_partitions/4*
  - *next_partition/3*
  - *next_partition/4*
  - *previous_partition/3*
  - *previous_partition/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

partitions/2

Generates all set partitions of a list using default order. Partitions are represented as lists of non-empty blocks. The empty list has one partition, represented by the empty list of blocks.

Compilation flags:
`static`

Template:
`partitions(List,Partitions)`
Mode and number of proofs:
`partitions(+list,-list)` - `one`

---

partition/2

True iff the second argument is a set partition of a list using default order.

Compilation flags:
`static`

Template:
`partition(List,Partition)`
Mode and number of proofs:
`partition(+list,-list)` - `one_or_more`

---

partitions/3

Generates either all set partitions with the given order (when the second argument is `default` or `lexicographic`) or all set partitions with an exact number of blocks using default order (when the first argument is an integer).

Compilation flags:
`static`

Template:
`partitions(ListOrK,OrderOrList,Partitions)`
Mode and number of proofs:
`partitions(+list,+atom,-list)` - `one`
`partitions(+integer,+list,-list)` - `one`

---

partition/3

True iff the third argument is a set partition generated either in the given order or with the given exact number of blocks using default order.

Compilation flags:
`static`

Template:
`partition(ListOrK,OrderOrList,Partition)`
Mode and number of proofs:
`partition(+list,+atom,-list)` - `one_or_more`
`partition(+integer,+list,-list)` - `one_or_more`

---

partitions/4

Generates all set partitions with an exact number of blocks and the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`partitions(K,List,Order,Partitions)`
Mode and number of proofs:
`partitions(+integer,+list,+atom,-list)` - `one`

---

partition/4

True iff the fourth argument is a set partition with an exact number of blocks and the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`partition(K,List,Order,Partition)`
Mode and number of proofs:
`partition(+integer,+list,+atom,-list)` - `one_or_more`

---

distinct_partitions/2

Generates all distinct set partitions of a list (deduplicating equal-valued partitions after canonicalizing block order) using default order.

Compilation flags:
`static`

Template:
`distinct_partitions(List,Partitions)`
Mode and number of proofs:
`distinct_partitions(+list,-list)` - `one`

---

distinct_partition/2

True iff the second argument is a distinct set partition of a list using default order.

Compilation flags:
`static`

Template:
`distinct_partition(List,Partition)`
Mode and number of proofs:
`distinct_partition(+list,-list)` - `one_or_more`

---

distinct_partitions/3

Generates either all distinct set partitions with the given order (when the second argument is `default` or `lexicographic`) or all distinct set partitions with an exact number of blocks using default order (when the first argument is an integer).

Compilation flags:
`static`

Template:
`distinct_partitions(ListOrK,OrderOrList,Partitions)`
Mode and number of proofs:
`distinct_partitions(+list,+atom,-list)` - `one`
`distinct_partitions(+integer,+list,-list)` - `one`

---

distinct_partition/3

True iff the third argument is a distinct set partition generated either in the given order or with the given exact number of blocks using default order.

Compilation flags:
`static`

Template:
`distinct_partition(ListOrK,OrderOrList,Partition)`
Mode and number of proofs:
`distinct_partition(+list,+atom,-list)` - `one_or_more`
`distinct_partition(+integer,+list,-list)` - `one_or_more`

---

distinct_partitions/4

Generates all distinct set partitions with an exact number of blocks and the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_partitions(K,List,Order,Partitions)`
Mode and number of proofs:
`distinct_partitions(+integer,+list,+atom,-list)` - `one`

---

distinct_partition/4

True iff the fourth argument is a distinct set partition with an exact number of blocks and the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_partition(K,List,Order,Partition)`
Mode and number of proofs:
`distinct_partition(+integer,+list,+atom,-list)` - `one_or_more`

---

count_partitions/2

Counts the number of set partitions of a list.

Compilation flags:
`static`

Template:
`count_partitions(List,Count)`
Mode and number of proofs:
`count_partitions(+list,-integer)` - `one`

---

nth_partition/3

Returns the set partition at a given zero-based index using default order.

Compilation flags:
`static`

Template:
`nth_partition(List,Index,Partition)`
Mode and number of proofs:
`nth_partition(+list,+integer,-list)` - `zero_or_one`

---

nth_partition/4

Returns either the set partition at a given zero-based index in the given order (when the second argument is `default` or `lexicographic`) or the set partition with an exact number of blocks at a given zero-based index using default order (when the first argument is an integer).

Compilation flags:
`static`

Template:
`nth_partition(ListOrK,OrderOrList,Index,Partition)`
Mode and number of proofs:
`nth_partition(+list,+atom,+integer,-list)` - `zero_or_one`
`nth_partition(+integer,+list,+integer,-list)` - `zero_or_one`

---

nth_partition/5

Returns the set partition with an exact number of blocks at a given zero-based index in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`nth_partition(K,List,Order,Index,Partition)`
Mode and number of proofs:
`nth_partition(+integer,+list,+atom,+integer,-list)` - `zero_or_one`

---

partition_index/3

Returns the zero-based index of a set partition using default order.

Compilation flags:
`static`

Template:
`partition_index(List,Partition,Index)`
Mode and number of proofs:
`partition_index(+list,+list,-integer)` - `zero_or_one`

---

partition_index/4

Returns either the zero-based index of a set partition in the given order (when the second argument is `default` or `lexicographic`) or the zero-based index of a set partition with an exact number of blocks using default order (when the first argument is an integer).

Compilation flags:
`static`

Template:
`partition_index(ListOrK,OrderOrList,Partition,Index)`
Mode and number of proofs:
`partition_index(+list,+atom,+list,-integer)` - `zero_or_one`
`partition_index(+integer,+list,+list,-integer)` - `zero_or_one`

---

partition_index/5

Returns the zero-based index of a set partition with an exact number of blocks in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`partition_index(K,List,Order,Partition,Index)`
Mode and number of proofs:
`partition_index(+integer,+list,+atom,+list,-integer)` - `zero_or_one`

---

count_partitions/3

Counts the number of set partitions of a list with an exact number of blocks.

Compilation flags:
`static`

Template:
`count_partitions(K,List,Count)`
Mode and number of proofs:
`count_partitions(+integer,+list,-integer)` - `zero_or_one`

---

count_distinct_partitions/2

Counts the number of distinct set partitions of a list (deduplicating equal-valued partitions).

Compilation flags:
`static`

Template:
`count_distinct_partitions(List,Count)`
Mode and number of proofs:
`count_distinct_partitions(+list,-integer)` - `one`

---

nth_distinct_partition/3

Returns the distinct set partition at a given zero-based index using default order.

Compilation flags:
`static`

Template:
`nth_distinct_partition(List,Index,Partition)`
Mode and number of proofs:
`nth_distinct_partition(+list,+integer,-list)` - `zero_or_one`

---

nth_distinct_partition/4

Returns either the distinct set partition at a given zero-based index in the given order (when the second argument is `default` or `lexicographic`) or the distinct set partition with an exact number of blocks at a given zero-based index using default order (when the first argument is an integer).

Compilation flags:
`static`

Template:
`nth_distinct_partition(ListOrK,OrderOrList,Index,Partition)`
Mode and number of proofs:
`nth_distinct_partition(+list,+atom,+integer,-list)` - `zero_or_one`
`nth_distinct_partition(+integer,+list,+integer,-list)` - `zero_or_one`

---

nth_distinct_partition/5

Returns the distinct set partition with an exact number of blocks at a given zero-based index in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`nth_distinct_partition(K,List,Order,Index,Partition)`
Mode and number of proofs:
`nth_distinct_partition(+integer,+list,+atom,+integer,-list)` - `zero_or_one`

---

distinct_partition_index/3

Returns the zero-based index of a distinct set partition using default order.

Compilation flags:
`static`

Template:
`distinct_partition_index(List,Partition,Index)`
Mode and number of proofs:
`distinct_partition_index(+list,+list,-integer)` - `zero_or_one`

---

distinct_partition_index/4

Returns either the zero-based index of a distinct set partition in the given order (when the second argument is `default` or `lexicographic`) or the zero-based index of a distinct set partition with an exact number of blocks using default order (when the first argument is an integer).

Compilation flags:
`static`

Template:
`distinct_partition_index(ListOrK,OrderOrList,Partition,Index)`
Mode and number of proofs:
`distinct_partition_index(+list,+atom,+list,-integer)` - `zero_or_one`
`distinct_partition_index(+integer,+list,+list,-integer)` - `zero_or_one`

---

distinct_partition_index/5

Returns the zero-based index of a distinct set partition with an exact number of blocks in the given order: `default` or `lexicographic`.

Compilation flags:
`static`

Template:
`distinct_partition_index(K,List,Order,Partition,Index)`
Mode and number of proofs:
`distinct_partition_index(+integer,+list,+atom,+list,-integer)` - `zero_or_one`

---

count_distinct_partitions/3

Counts the number of distinct set partitions of a list with an exact number of blocks.

Compilation flags:
`static`

Template:
`count_distinct_partitions(K,List,Count)`
Mode and number of proofs:
`count_distinct_partitions(+integer,+list,-integer)` - `zero_or_one`

---

random_partition/2

Returns a random set partition of a list.

Compilation flags:
`static`

Template:
`random_partition(List,Partition)`
Mode and number of proofs:
`random_partition(+list,-list)` - `one`

---

random_partition/3

Returns a random set partition of a list with an exact number of blocks.

Compilation flags:
`static`

Template:
`random_partition(K,List,Partition)`
Mode and number of proofs:
`random_partition(+integer,+list,-list)` - `zero_or_one`

---

sample_partitions/3

Returns SampleCount random set partitions of a list, sampled with replacement.

Compilation flags:
`static`

Template:
`sample_partitions(List,SampleCount,Samples)`
Mode and number of proofs:
`sample_partitions(+list,+integer,-list)` - `zero_or_one`

---

sample_partitions/4

Returns SampleCount random set partitions of a list with an exact number of blocks, sampled with replacement.

Compilation flags:
`static`

Template:
`sample_partitions(K,List,SampleCount,Samples)`
Mode and number of proofs:
`sample_partitions(+integer,+list,+integer,-list)` - `zero_or_one`

---

random_distinct_partition/2

Returns a random distinct set partition of a list (deduplicating equal-valued partitions).

Compilation flags:
`static`

Template:
`random_distinct_partition(List,Partition)`
Mode and number of proofs:
`random_distinct_partition(+list,-list)` - `one`

---

random_distinct_partition/3

Returns a random distinct set partition of a list with an exact number of blocks (deduplicating equal-valued partitions).

Compilation flags:
`static`

Template:
`random_distinct_partition(K,List,Partition)`
Mode and number of proofs:
`random_distinct_partition(+integer,+list,-list)` - `zero_or_one`

---

sample_distinct_partitions/3

Returns SampleCount random distinct set partitions of a list, sampled with replacement after deduplicating equal-valued partitions.

Compilation flags:
`static`

Template:
`sample_distinct_partitions(List,SampleCount,Samples)`
Mode and number of proofs:
`sample_distinct_partitions(+list,+integer,-list)` - `zero_or_one`

---

sample_distinct_partitions/4

Returns SampleCount random distinct set partitions of a list with an exact number of blocks, sampled with replacement after deduplicating equal-valued partitions.

Compilation flags:
`static`

Template:
`sample_distinct_partitions(K,List,SampleCount,Samples)`
Mode and number of proofs:
`sample_distinct_partitions(+integer,+list,+integer,-list)` - `zero_or_one`

---

next_partition/3

Returns the next distinct set partition value in lexicographic order induced by the first argument.

Compilation flags:
```
static
```

Template:
```
next_partition(List,Partition,Next)
```
Mode and number of proofs:
```
next_partition(+list,+list,-list) - zero_or_one
```

---

next_partition/4

Returns the next distinct set partition value with an exact number of blocks in lexicographic order induced by the second argument.

Compilation flags:
```
static
```

Template:
```
next_partition(K,List,Partition,Next)
```
Mode and number of proofs:
```
next_partition(+integer,+list,+list,-list) - zero_or_one
```

---

previous_partition/3

Returns the previous distinct set partition value in lexicographic order induced by the first argument.

Compilation flags:
```
static
```

Template:
```
previous_partition(List,Partition,Previous)
```
Mode and number of proofs:
```
previous_partition(+list,+list,-list) - zero_or_one
```

---

previous_partition/4

Returns the previous distinct set partition value with an exact number of blocks in lexicographic order induced by the second argument.

Compilation flags:
`static`

Template:
`previous_partition(K,List,Partition,Previous)`
Mode and number of proofs:
`previous_partition(+integer,+list,+list,-list)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.252 paseto

object

1.252.1 paseto

Facade for PASETO v4 JSON claims encryption, signing, authentication, validation, and key selection.

Availability:
`logtalk_load(paseto(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-28

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *paseto_helpers*
`public` *paseto_claims_helpers*
Uses:
list

paseto_claims
paseto_keys
paseto_v4

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *encrypt/4*
  - *decrypt/4*
  - *decrypt/5*
  - *sign/4*
  - *verify/4*
  - *verify/5*
  - *claims/2*
  - *claim/3*
  - *validate_claims/3*
  - *validate_claim/3*
  - *peek_key_id/2*
  - *validate_key_set/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

encrypt/4

Encrypts a JSON claims object as a v4.local token.

Compilation flags:
`static`

Template:
`encrypt(Claims,Key,Token,Options)`
Mode and number of proofs:
`encrypt(+term,+list(byte),-atom,+list(compound))` - `one_or_error`

Exceptions:
`Claims` is not a JSON object or contains duplicate members:
`domain_error(paseto_json_object,Claims)`
`Key` is not a list of 32 bytes:
`type_error(list(byte,32),Key)`
The footer `kid` conflicts with the `key_id/1` option:

`domain_error(paseto_footer_key_id,Existing-KeyId)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

---

decrypt/4

Authenticates and decrypts a v4.local claims token using a key or key set.

Compilation flags:

`static`

Template:

`decrypt(Token,KeyOrKeySet,Claims,Options)`

Mode and number of proofs:

`decrypt(+atom,+term,-term,+list(compound))` - `zero_or_one_or_error`

Exceptions:

`Token` is not a canonical v4.local token:

`domain_error(paseto_v4_token,Token)`

`KeyOrKeySet` is an invalid key set:

`domain_error(paseto_key_set,KeyOrKeySet)`

No key can be selected from `KeyOrKeySet`:

`existence_error(paseto_key,local-KeyId)`

The authenticated payload is not a JSON object:

`domain_error(paseto_json_object,Claims)`

The claims do not satisfy the validation policy:

`domain_error(paseto_claims,Reason)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

---

decrypt/5

Authenticates and decrypts a v4.local claims token and returns its authenticated footer JSON object.

Compilation flags:
`static`

Template:
`decrypt(Token,KeyOrKeySet,Claims,Footer,Options)`
Mode and number of proofs:
`decrypt(+atom,+term,-term,-term,+list(compound))` - `zero_or_one_or_error`

Exceptions:
`Token` is not a canonical v4.local token:
`domain_error(paseto_v4_token,Token)`
`KeyOrKeySet` is an invalid key set:
`domain_error(paseto_key_set,KeyOrKeySet)`
No key can be selected from `KeyOrKeySet`:
`existence_error(paseto_key,local-KeyId)`
The authenticated payload or footer is not a JSON object:
`domain_error(paseto_json_object,JSON)`
The claims do not satisfy the validation policy:
`domain_error(paseto_claims,Reason)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

sign/4

Signs a JSON claims object as a v4.public token using an Ed25519 seed.

Compilation flags:
`static`

Template:
`sign(Claims,Seed,Token,Options)`
Mode and number of proofs:
`sign(+term,+list(byte),-atom,+list(compound))` - `one_or_error`

Exceptions:

`Claims` is not a JSON object or contains duplicate members:
```
domain_error(paseto_json_object,Claims)
```
`Seed` is not a list of 32 bytes:
```
type_error(list(byte,32),Seed)
```
The footer `kid` conflicts with the `key_id/1` option:
```
domain_error(paseto_footer_key_id,Existing-KeyId)
```
`Options` is a variable:
```
instantiation_error
```
`Options` is neither a variable nor a list:
```
type_error(list,Options)
```
An element `Option` of the list `Options` is a variable:
```
instantiation_error
```
An element `Option` of the list `Options` is neither a variable nor a compound term:
```
type_error(compound,Option)
```
An element `Option` of the list `Options` is a compound term but not a valid option:
```
domain_error(option,Option)
```

---

verify/4

Authenticates and validates a v4.public claims token using a public key or key set.

Compilation flags:
```
static
```

Template:
```
verify(Token,KeyOrKeySet,Claims,Options)
```
Mode and number of proofs:
```
verify(+atom,+term,-term,+list(compound)) - zero_or_one_or_error
```

Exceptions:
`Token` is not a canonical v4.public token:
```
domain_error(paseto_v4_token,Token)
```
`KeyOrKeySet` is an invalid key set:
```
domain_error(paseto_key_set,KeyOrKeySet)
```
No key can be selected from `KeyOrKeySet`:
```
existence_error(paseto_key,(public)-KeyId)
```
The authenticated payload is not a JSON object:
```
domain_error(paseto_json_object,Claims)
```
The claims do not satisfy the validation policy:
```
domain_error(paseto_claims,Reason)
```
`Options` is a variable:
```
instantiation_error
```
`Options` is neither a variable nor a list:
```
type_error(list,Options)
```
An element `Option` of the list `Options` is a variable:
```
instantiation_error
```
An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

`verify/5`

Authenticates and validates a v4.public claims token and returns its authenticated footer JSON object.

Compilation flags:
`static`

Template:
`verify(Token,KeyOrKeySet,Claims,Footer,Options)`
Mode and number of proofs:
`verify(+atom,+term,-term,-term,+list(compound))` - `zero_or_one_or_error`

Exceptions:
`Token` is not a canonical v4.public token:
`domain_error(paseto_v4_token,Token)`
`KeyOrKeySet` is an invalid key set:
`domain_error(paseto_key_set,KeyOrKeySet)`
No key can be selected from `KeyOrKeySet`:
`existence_error(paseto_key,(public)-KeyId)`
The authenticated payload or footer is not a JSON object:
`domain_error(paseto_json_object,JSON)`
The claims do not satisfy the validation policy:
`domain_error(paseto_claims,Reason)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

claims/2

Decodes claims from a v4.public token without authenticating them. Rejects local tokens. The result must not be trusted.

Compilation flags:
`static`

Template:
`claims(Token,Claims)`
Mode and number of proofs:
`claims(+atom,-term)` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is not a canonical v4.public token:
`domain_error(paseto_v4_token,Token)`
`Token` has a malformed public payload:
`domain_error(paseto_v4_public_payload,Token)`
`Token` payload is not a JSON object:
`domain_error(paseto_json_object,Claims)`

---

claim/3

Looks up a value in a claims object.

Compilation flags:
`static`

Template:
`claim(Claims,Name,Value)`
Mode and number of proofs:
`claim(+term,+atom,-term)` - `zero_or_one`

---

validate_claims/3

Validates a claims object using a policy list and options.

Compilation flags:
`static`

Template:
`validate_claims(Claims,Policy,Options)`

Mode and number of proofs:

`validate_claims(+term,+list(compound),+list(compound))` - `one_or_error`

Exceptions:

`Claims` is not a JSON object or contains duplicate members:

`domain_error(paseto_json_object,Claims)`

`Claims` is missing a required claim `Name`:

`domain_error(paseto_claims,missing(Name))`

`Policy` contains an invalid claim policy:

`domain_error(paseto_claim_policy,ClaimPolicy)`

A time claim has a non-numeric value:

`type_error(time_number,Name-Value)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`validate_claim/3`

Validates one claim policy.

Compilation flags:

`static`

Template:

`validate_claim(Claims,ClaimPolicy,Options)`

Meta-predicate template:

`validate_claim(*,*,*)`

Mode and number of proofs:

`validate_claim(+term,+compound,+list(compound))` - `one_or_error`

Exceptions:

`Claims` is missing the required claim `Name`:

`domain_error(paseto_claims,missing(Name))`

The value of claim `Name` does not satisfy the policy:

`domain_error(paseto_claim(Name),Value)`

`ClaimPolicy` is not a supported claim policy:

`domain_error(paseto_claim_policy,ClaimPolicy)`

A time claim has an unknown validation kind:

`domain_error(paseto_time_claim_kind,Kind)`

A time claim has a non-numeric value:

`type_error(time_number,Name-Value)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

---

peek_key_id/2

Reads a kid value from the unauthenticated token footer for key selection. The result must not be trusted.

Compilation flags:

`static`

Template:

`peek_key_id(Token,KeyId)`

Mode and number of proofs:

`peek_key_id(+atom,-atom)` - `zero_or_one_or_error`

Exceptions:

`Token` is a variable:

`instantiation_error`

`Token` is neither a variable nor an atom:

`type_error(atom,Token)`

`Token` is not a canonical v4 token:

`domain_error(paseto_v4_token,Token)`

`Token` footer is not a JSON object:

`domain_error(paseto_json_object,Footer)`

---

validate_key_set/1

Validates a native PASETO key_set/1 term.

Compilation flags:

`static`

Template:

`validate_key_set(KeySet)`

Mode and number of proofs:

`validate_key_set(+compound)` - `one_or_error`

Exceptions:
`KeySet` is not a valid `key_set/1` term:
`domain_error(paseto_key_set,KeySet)`
`KeySet` contains an invalid key record `Record`:
`domain_error(paseto_key_record,Record)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.252.2 `paseto_claims`

PASETO JSON claims lookup and validation predicates.

Availability:
`logtalk_load(paseto(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *paseto_claims_helpers*
Uses:
list
os

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*

    - *validate_claims/3*
    - *validate_claim/3*
    - *claim/3*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

validate_claims/3

Validates claims using registered-claim defaults, a policy list, and options.

Compilation flags:

`static`

Template:

`validate_claims(Claims,Policy,Options)`

Mode and number of proofs:

`validate_claims(+term,+list(compound),+list(compound))` - `one_or_error`

Exceptions:

`Claims` is not a JSON object or contains duplicate members:

`domain_error(paseto_json_object,Claims)`

`Claims` is missing a required claim `Name`:

`domain_error(paseto_claims,missing(Name))`

`Policy` contains an invalid claim policy:

`domain_error(paseto_claim_policy,ClaimPolicy)`

A time claim has a non-numeric value:

`type_error(time_number,Name-Value)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

validate_claim/3

Validates one claim policy against a claims object.

Compilation flags:
`static`

Template:
`validate_claim(Claims,ClaimPolicy,Options)`
Meta-predicate template:
`validate_claim(*,*,*)`
Mode and number of proofs:
`validate_claim(+term,+compound,+list(compound))` - `one_or_error`

Exceptions:
`Claims` is missing the required claim `Name`:
`domain_error(paseto_claims,missing(Name))`
The value of claim `Name` does not satisfy the policy:
`domain_error(paseto_claim(Name),Value)`
`ClaimPolicy` is not a supported claim policy:
`domain_error(paseto_claim_policy,ClaimPolicy)`
A time claim has an unknown validation kind:
`domain_error(paseto_time_claim_kind,Kind)`
A time claim has a non-numeric value:
`type_error(time_number,Name-Value)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

claim/3

Looks up a claim value by name.

Compilation flags:
`static`

Template:
`claim(Claims,Name,Value)`
Mode and number of proofs:
`claim(+term,+atom,-term)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.252.3 paseto_claims_helpers

Shared JSON and option helpers for PASETO claims workflows.

Availability:
`logtalk_load(paseto(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static`

Extends:
`public` *options*
Uses:
json
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *json_member/3*
  - *json_object_pairs/2*
  - *json_object/1*
  - *json_bytes/2*
  - *byte_list/1*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

json_member/3

Enumerates JSON object member values for a key.

Compilation flags:
`static`

Template:
`json_member(Key,Object,Value)`
Mode and number of proofs:
`json_member(+atom,+term,-term)` - `zero_or_more`

---

json_object_pairs/2

Extracts pairs from a JSON object term.

Compilation flags:
`static`

Template:
`json_object_pairs(Object,Pairs)`
Mode and number of proofs:
`json_object_pairs(+term,-list(compound))` - `zero_or_one`

---

json_object/1

Validates a JSON object and rejects duplicate member names.

Compilation flags:
`static`

Template:
`json_object(Object)`
Mode and number of proofs:
`json_object(+term)` - `one_or_error`

Exceptions:
`Object` is not a JSON object or contains duplicate members:
`domain_error(paseto_json_object,Object)`

---

json_bytes/2

Converts between a JSON term and its UTF-8-compatible byte representation.

Compilation flags:
`static`

Template:
`json_bytes(JSON,Bytes)`
Mode and number of proofs:
`json_bytes(+term,-list(byte))` - `one_or_error`
`json_bytes(-term,+list(byte))` - `one_or_error`

Exceptions:
`JSON` cannot be generated as JSON text:
`domain_error(json_term,JSON)`
`Bytes` cannot be parsed as JSON text:
`domain_error(json_text,Bytes)`

---

byte_list/1

Succeeds when the argument is a proper list of bytes.

Compilation flags:
`static`

Template:
`byte_list(Bytes)`
Mode and number of proofs:
`byte_list(+term)` - `zero_or_one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.252.4 paseto_helpers

Protected PASETO v4 framing, PAE, and encoding helpers.

Availability:
`logtalk_load(paseto(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static`

Uses:
base64url_no_padding
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *pae/2*
  - *parse_token/4*
  - *format_token/4*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

pae/2

Pre-authentication encodes a list of byte-list pieces.

Compilation flags:
`static`

Template:
`pae(Pieces,Encoding)`
Mode and number of proofs:

```
pae(+list(list(byte)),-list(byte)) - one
```

---

parse_token/4

Strictly parses a canonical v4 token for Purpose and decodes its body and optional footer.

Compilation flags:

```
static
```

Template:

```
parse_token(Token,Purpose,Body,Footer)
```

Mode and number of proofs:

```
parse_token(+atom,+atom,-list(byte),-list(byte)) - one_or_error
```

Exceptions:

`Token` is a variable:

```
instantiation_error
```

`Token` is neither a variable nor an atom:

```
type_error(atom,Token)
```

`Token` is an atom but does not have the expected PASETO v4 header:

```
domain_error(paseto_v4_token,Token)
```

`Token` is a PASETO v4 header but has a malformed compact serialization:

```
domain_error(paseto_compact_serialization,malformed)
```

`Token` contains non-canonical base64url data:

```
representation_error(base64)
```

---

format_token/4

Formats a canonical v4 token for Purpose from body and footer bytes.

Compilation flags:

```
static
```

Template:

```
format_token(Purpose,Body,Footer,Token)
```

Mode and number of proofs:

```
format_token(+atom,+list(byte),+list(byte),-atom) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.252.5 paseto_keys

Validation and selection for native PASETO v4 key sets.

Availability:
`logtalk_load(paseto(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-28

Compilation flags:
`static, context_switching_calls`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *validate/1*
  - *select_keys/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

validate/1

Validates a key_set/1 term containing local/2 and public/2 records.

Compilation flags:
`static`

Template:

```
validate(KeySet)
```

Mode and number of proofs:

```
validate(+compound) - one_or_error
```

Exceptions:

`KeySet` is not a valid `key_set/1` term:

```
domain_error(paseto_key_set,KeySet)
```

`KeySet` contains an invalid key record `Record`:

```
domain_error(paseto_key_record,Record)
```

`select_keys/4`

Selects all keys matching Purpose and optional KeyId, preserving key-set order.

Compilation flags:

```
static
```

Template:

```
select_keys(KeySet,Purpose,KeyId,Keys)
```

Mode and number of proofs:

```
select_keys(+compound,+atom,+term,-list(list(byte))) - one_or_error
```

Exceptions:

`KeySet` is not a valid `key_set/1` term:

```
domain_error(paseto_key_set,KeySet)
```

`KeySet` contains an invalid key record `Record`:

```
domain_error(paseto_key_record,Record)
```

No key matches `Purpose` and `KeyId`:

```
existence_error(paseto_key,Purpose-KeyId)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.252.6 paseto_protocol

PASETO v4.local and v4.public protocol for byte payloads.

Availability:
`logtalk_load(paseto(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *local_key/1*
  - *public_keypair/2*
  - *local_encrypt/3*
  - *local_encrypt/5*
  - *local_decrypt/3*
  - *local_decrypt/5*
  - *public_sign/3*
  - *public_sign/5*
  - *public_verify/3*
  - *public_verify/5*
  - *footer/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

local_key/1

Generates a 32-byte v4.local key.

Compilation flags:
`static`

Template:
`local_key(Key)`
Mode and number of proofs:
`local_key(-list(byte))` - `one`

---

public_keypair/2

Generates an Ed25519 seed and public key for v4.public tokens.

Compilation flags:
`static`

Template:
`public_keypair(Seed,PublicKey)`
Mode and number of proofs:
`public_keypair(-list(byte),-list(byte))` - `one`

---

local_encrypt/3

Encrypts Payload using a 32-byte local Key and empty footer and implicit assertion.

Compilation flags:
`static`

Template:
`local_encrypt(Key,Payload,Token)`
Mode and number of proofs:
`local_encrypt(+list(byte),+list(byte),-atom)` - `one_or_error`

Exceptions:
`Key` is a variable or a partial list:
`instantiation_error`
`Key` is not a list of the required number of bytes:
`type_error(list(byte,32),Key)`
`Key` contains a non-integer byte:
`type_error(integer,Byte)`

`Key` contains an integer outside the byte range:
```
domain_error(byte,Byte)
```
`Payload` is a variable or a partial list:
```
instantiation_error
```
`Payload` is not a list of the required number of bytes:
```
type_error(list(byte,32),Payload)
```
`Payload` contains a non-integer byte:
```
type_error(integer,Byte)
```
`Payload` contains an integer outside the byte range:
```
domain_error(byte,Byte)
```

`local_encrypt/5`

Encrypts Payload using a 32-byte local Key and authenticates Footer and ImplicitAssertion.

Compilation flags:
```
static
```

Template:
```
local_encrypt(Key,Payload,Footer,ImplicitAssertion,Token)
```
Mode and number of proofs:
```
local_encrypt(+list(byte),+list(byte),+list(byte),+list(byte),-atom) -
one_or_error
```

Exceptions:
`Key` is a variable or a partial list:
```
instantiation_error
```
`Key` is not a list of the required number of bytes:
```
type_error(list(byte,32),Key)
```
`Key` contains a non-integer byte:
```
type_error(integer,Byte)
```
`Key` contains an integer outside the byte range:
```
domain_error(byte,Byte)
```
`Payload` is a variable or a partial list:
```
instantiation_error
```
`Payload` is not a list of the required number of bytes:
```
type_error(list(byte,32),Payload)
```
`Payload` contains a non-integer byte:
```
type_error(integer,Byte)
```
`Payload` contains an integer outside the byte range:
```
domain_error(byte,Byte)
```
`Footer` is a variable or a partial list:
```
instantiation_error
```
`Footer` is not a list of the required number of bytes:
```
type_error(list(byte,32),Footer)
```
`Footer` contains a non-integer byte:
```
type_error(integer,Byte)
```

`Footer` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`ImplicitAssertion` is a variable or a partial list:
`instantiation_error`
`ImplicitAssertion` is not a list of the required number of bytes:
`type_error(list(byte,32),ImplicitAssertion)`
`ImplicitAssertion` contains a non-integer byte:
`type_error(integer,Byte)`
`ImplicitAssertion` contains an integer outside the byte range:
`domain_error(byte,Byte)`

`local_decrypt/3`

Authenticates and decrypts a local Token using an empty implicit assertion.

Compilation flags:
`static`

Template:
`local_decrypt(Token,Key,Payload)`
Mode and number of proofs:
`local_decrypt(+atom,+list(byte),-list(byte))` - `zero_or_one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a canonical v4.local token:
`domain_error(paseto_v4_token,Token)`
`Key` is a variable or a partial list:
`instantiation_error`
`Key` is not a list of the required number of bytes:
`type_error(list(byte,32),Key)`
`Key` contains a non-integer byte:
`type_error(integer,Byte)`
`Key` contains an integer outside the byte range:
`domain_error(byte,Byte)`

---

local_decrypt/5

Authenticates and decrypts a local Token using ImplicitAssertion and returns its authenticated Footer.

Compilation flags:

```
static
```

Template:

```
local_decrypt(Token,Key,ImplicitAssertion,Payload,Footer)
```

Mode and number of proofs:

```
local_decrypt(+atom,+list(byte),+list(byte),-list(byte),-list(byte)) -
zero_or_one_or_error
```

Exceptions:

`Token` is a variable:

```
instantiation_error
```

`Token` is neither a variable nor an atom:

```
type_error(atom,Token)
```

`Token` is an atom but not a canonical v4.local token:

```
domain_error(paseto_v4_token,Token)
```

`Token` is a v4.local token but has a malformed compact serialization:

```
domain_error(paseto_compact_serialization,malformed)
```

`Token` contains non-canonical base64url data:

```
representation_error(base64)
```

`Token` has a malformed local payload:

```
domain_error(paseto_v4_local_payload,Token)
```

`Key` is a variable or a partial list:

```
instantiation_error
```

`Key` is not a list of the required number of bytes:

```
type_error(list(byte,32),Key)
```

`Key` contains a non-integer byte:

```
type_error(integer,Byte)
```

`Key` contains an integer outside the byte range:

```
domain_error(byte,Byte)
```

`ImplicitAssertion` is a variable or a partial list:

```
instantiation_error
```

`ImplicitAssertion` is not a list of the required number of bytes:

```
type_error(list(byte,32),ImplicitAssertion)
```

`ImplicitAssertion` contains a non-integer byte:

```
type_error(integer,Byte)
```

`ImplicitAssertion` contains an integer outside the byte range:

```
domain_error(byte,Byte)
```

public_sign/3

Signs Payload using an Ed25519 Seed and empty footer and implicit assertion.

Compilation flags:
`static`

Template:
`public_sign(Seed,Payload,Token)`
Mode and number of proofs:
`public_sign(+list(byte),+list(byte),-atom)` - `one_or_error`

Exceptions:
`Seed` is a variable or a partial list:
`instantiation_error`
`Seed` is not a list of the required number of bytes:
`type_error(list(byte,32),Seed)`
`Seed` contains a non-integer byte:
`type_error(integer,Byte)`
`Seed` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Payload` is a variable or a partial list:
`instantiation_error`
`Payload` is not a list of the required number of bytes:
`type_error(list(byte,32),Payload)`
`Payload` contains a non-integer byte:
`type_error(integer,Byte)`
`Payload` contains an integer outside the byte range:
`domain_error(byte,Byte)`

---

public_sign/5

Signs Payload using an Ed25519 Seed and authenticates Footer and ImplicitAssertion.

Compilation flags:
`static`

Template:
`public_sign(Seed,Payload,Footer,ImplicitAssertion,Token)`
Mode and number of proofs:
`public_sign(+list(byte),+list(byte),+list(byte),+list(byte),-atom)` - `one_or_error`

Exceptions:
`Seed` is a variable or a partial list:
`instantiation_error`
`Seed` is not a list of the required number of bytes:
`type_error(list(byte,32),Seed)`

`Seed` contains a non-integer byte:
`type_error(integer,Byte)`
`Seed` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Payload` is a variable or a partial list:
`instantiation_error`
`Payload` is not a list of the required number of bytes:
`type_error(list(byte,32),Payload)`
`Payload` contains a non-integer byte:
`type_error(integer,Byte)`
`Payload` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`Footer` is a variable or a partial list:
`instantiation_error`
`Footer` is not a list of the required number of bytes:
`type_error(list(byte,32),Footer)`
`Footer` contains a non-integer byte:
`type_error(integer,Byte)`
`Footer` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`ImplicitAssertion` is a variable or a partial list:
`instantiation_error`
`ImplicitAssertion` is not a list of the required number of bytes:
`type_error(list(byte,32),ImplicitAssertion)`
`ImplicitAssertion` contains a non-integer byte:
`type_error(integer,Byte)`
`ImplicitAssertion` contains an integer outside the byte range:
`domain_error(byte,Byte)`

`public_verify/3`

Authenticates a public Token using an Ed25519 public key and empty implicit assertion.

Compilation flags:
`static`

Template:
`public_verify(Token,PublicKey,Payload)`
Mode and number of proofs:
`public_verify(+atom,+list(byte),-list(byte))` - `zero_or_one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a canonical v4.public token:

`domain_error(paseto_v4_token,Token)`
`PublicKey` is a variable or a partial list:
`instantiation_error`
`PublicKey` is not a list of the required number of bytes:
`type_error(list(byte,32),PublicKey)`
`PublicKey` contains a non-integer byte:
`type_error(integer,Byte)`
`PublicKey` contains an integer outside the byte range:
`domain_error(byte,Byte)`

---

public_verify/5

Authenticates a public Token using an Ed25519 public key and ImplicitAssertion and returns its authenticated Footer.

Compilation flags:
`static`

Template:
`public_verify(Token,PublicKey,ImplicitAssertion,Payload,Footer)`
Mode and number of proofs:
`public_verify(+atom,+list(byte),+list(byte),-list(byte),-list(byte))` - `zero_or_one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a canonical v4 token:
`domain_error(paseto_v4_token,Token)`
`Token` is a canonical v4 token but has a malformed compact serialization:
`domain_error(paseto_compact_serialization,malformed)`
`Token` contains non-canonical base64url data:
`representation_error(base64)`
`Token` has a malformed public payload:
`domain_error(paseto_v4_public_payload,Token)`
`PublicKey` is a variable or a partial list:
`instantiation_error`
`PublicKey` is not a list of the required number of bytes:
`type_error(list(byte,32),PublicKey)`
`PublicKey` contains a non-integer byte:
`type_error(integer,Byte)`
`PublicKey` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`ImplicitAssertion` is a variable or a partial list:
`instantiation_error`
`ImplicitAssertion` is not a list of the required number of bytes:

`type_error(list(byte,32),ImplicitAssertion)`
`ImplicitAssertion` contains a non-integer byte:
`type_error(integer,Byte)`
`ImplicitAssertion` contains an integer outside the byte range:
`domain_error(byte,Byte)`

`footer/2`

Extracts a token footer without authenticating it. The result must only be used for pre-authentication key selection.

Compilation flags:
`static`

Template:
`footer(Token,Footer)`
Mode and number of proofs:
`footer(+atom,-list(byte))` - `one_or_error`

Exceptions:
`Token` is a variable:
`instantiation_error`
`Token` is neither a variable nor an atom:
`type_error(atom,Token)`
`Token` is an atom but not a canonical v4 token:
`domain_error(paseto_v4_token,Token)`
`Token` is a canonical v4 token but has a malformed compact serialization:
`domain_error(paseto_compact_serialization,malformed)`
`Token` contains non-canonical base64url data:
`representation_error(base64)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.252.7 paseto_v4

PASETO v4.local and v4.public implementation for byte payloads.

Availability:
`logtalk_load(paseto(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *paseto_protocol*
Imports:
`public` *paseto_helpers*
Uses:
blake2b(Key,DigestSize)
crypto
list
type

Remarks:
(none)

Inherited public predicates:
footer/2 *local_decrypt/3* *local_decrypt/5* *local_encrypt/3* *local_encrypt/5* *local_key/1* *public_keypair/2* *public_sign/3* *public_sign/5* *public_verify/3* *public_verify/5*

- *Public predicates*
- *Protected predicates*
  - *local_encrypt_with_nonce/6*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

`local_encrypt_with_nonce/6`

Encrypts a local payload using an explicit 32-byte nonce. Intended for deterministic conformance tests; production callers must use local_encrypt/3 or local_encrypt/5.

Compilation flags:

`static`

Template:

`local_encrypt_with_nonce(Key,Nonce,Payload,Footer,ImplicitAssertion,Token)`

Mode and number of proofs:

`local_encrypt_with_nonce(+list(byte),+list(byte),+list(byte),+list(byte),+list(byte),-atom)` - `one_or_error`

Exceptions:

`Key` is a variable or a partial list:

`instantiation_error`

`Key` is not a list of the required number of bytes:

`type_error(list(byte,32),Key)`

`Key` contains a non-integer byte:

`type_error(integer,Byte)`

`Key` contains an integer outside the byte range:

`domain_error(byte,Byte)`

`Nonce` is a variable or a partial list:

`instantiation_error`

`Nonce` is not a list of the required number of bytes:

`type_error(list(byte,32),Nonce)`

`Nonce` contains a non-integer byte:

`type_error(integer,Byte)`

`Nonce` contains an integer outside the byte range:

`domain_error(byte,Byte)`

`Payload` is a variable or a partial list:

`instantiation_error`

`Payload` is not a list of the required number of bytes:

`type_error(list(byte,32),Payload)`

`Payload` contains a non-integer byte:

`type_error(integer,Byte)`

`Payload` contains an integer outside the byte range:

`domain_error(byte,Byte)`

`Footer` is a variable or a partial list:

`instantiation_error`

`Footer` is not a list of the required number of bytes:

`type_error(list(byte,32),Footer)`

`Footer` contains a non-integer byte:

`type_error(integer,Byte)`

`Footer` contains an integer outside the byte range:
`domain_error(byte,Byte)`
`ImplicitAssertion` is a variable or a partial list:
`instantiation_error`
`ImplicitAssertion` is not a list of the required number of bytes:
`type_error(list(byte,32),ImplicitAssertion)`
`ImplicitAssertion` contains a non-integer byte:
`type_error(integer,Byte)`
`ImplicitAssertion` contains an integer outside the byte range:
`domain_error(byte,Byte)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.253 pattern_mining_protocols

category

1.253.1 `pattern_miner_common`

Shared predicates for pattern miner diagnostics, defaults, option handling, and export helpers.

Availability:
`logtalk_load(pattern_mining_protocols(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-06-07

Compilation flags:
`static`

Implements:
`public` *pattern_miner_protocol*
Extends:
`public` *options*
Uses:
format
list
type

Remarks:

(none)

Inherited public predicates:

check_option/1 check_options/1 check_pattern_miner/1 default_option/1 default_options/1 diagnostic/2 diagnostics/2 export_to_clauses/4 export_to_file/4 mine/2 mine/3 option/2 option/3 pattern_miner_options/2 print_pattern_miner/1 valid_option/1 valid_options/1 valid_pattern_miner/1

- *Public predicates*
- *Protected predicates*
  - *pattern_miner_diagnostics_data/2*
  - *pattern_miner_diagnostics/6*
  - *pattern_miner_export_template/4*
  - *valid_pattern_miner_metadata/5*
  - *check_item_domain/1*
  - *effective_support_count/3*
  - *effective_maximum_pattern_length/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

pattern_miner_diagnostics_data/2

Hook predicate that importing pattern miner implementations must define in order to expose diagnostics metadata.

Compilation flags:

```
static
```

Template:

```
pattern_miner_diagnostics_data(PatternMiner,Diagnostics)
```

Mode and number of proofs:

```
pattern_miner_diagnostics_data(+compound,-list(compound)) - one
```

---

pattern_miner_diagnostics/6

Builds a standard diagnostics list from generic pattern miner metadata and algorithm-specific diagnostics terms.

Compilation flags:

```
static
```

Template:

```
pattern_miner_diagnostics(Model,ItemDomain,Patterns,Options,SpecificDiagnostics,
Diagnostics)
```

Mode and number of proofs:

```
pattern_miner_diagnostics(+atom,+list(atom),+list(compound),+list(compound),
+list(compound),-list(compound)) - one
```

---

pattern_miner_export_template/4

Hook predicate that importing pattern miner implementations must define in order to expose the exported pattern miner template for a given functor.

Compilation flags:

```
static
```

Template:

```
pattern_miner_export_template(Dataset,PatternMiner,Functor,Template)
```

Mode and number of proofs:

```
pattern_miner_export_template(+object_identifier,+compound,+atom,-callable) - one
```

---

valid_pattern_miner_metadata/5

True when the diagnostics list matches the generic pattern miner metadata implied by the model, item domain, mined patterns, and effective options.

Compilation flags:

```
static
```

Template:

```
valid_pattern_miner_metadata(Model,ItemDomain,Patterns,Options,Diagnostics)
```

Mode and number of proofs:

```
valid_pattern_miner_metadata(+atom,+list(atom),+list(compound),+list(compound),
+list(compound)) - zero_or_one
```

---

check_item_domain/1

Validates that the declared item domain is canonical and contains only atoms.

Compilation flags:

```
static
```

Template:

```
check_item_domain(ItemDomain)
```

Mode and number of proofs:

```
check_item_domain(+list(atom)) - one
```

---

effective_support_count/3

Computes the effective minimum support count from the effective options and the dataset size.

Compilation flags:
```
static
```

Template:
```
effective_support_count(DatasetSize,Options,SupportCount)
```
Mode and number of proofs:
```
effective_support_count(+integer,+list(compound),-integer) - one
```

---

effective_maximum_pattern_length/3

Computes the effective maximum pattern length from the effective options and the dataset maximum pattern length.

Compilation flags:
```
static
```

Template:
```
effective_maximum_pattern_length(DatasetMaximumPatternLength,Options,
MaximumPatternLength)
```
Mode and number of proofs:
```
effective_maximum_pattern_length(+integer,+list(compound),-integer) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.253.2 pattern_miner_protocol

Protocol for machine learning pattern-mining algorithms.

Availability:
```
logtalk_load(pattern_mining_protocols(loader))
```

Author: Paulo Moura

Version: 1:0:0
Date: 2026-04-30

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *mine/3*
  - *mine/2*
  - *diagnostics/2*
  - *diagnostic/2*
  - *pattern_miner_options/2*
  - *check_pattern_miner/1*
  - *valid_pattern_miner/1*
  - *export_to_clauses/4*
  - *export_to_file/4*
  - *print_pattern_miner/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

mine/3

Mines a pattern miner result from the given dataset object using the specified options.

Compilation flags:
`static`

Template:
`mine(Dataset,PatternMiner,Options)`
Mode and number of proofs:
`mine(+object_identifier,-compound,+list(compound))` - `one`

mine/2

Mines a pattern miner result from the given dataset object using default options.

Compilation flags:
```
static
```

Template:
```
mine(Dataset,PatternMiner)
```
Mode and number of proofs:
```
mine(+object_identifier,-compound) - one
```

---

diagnostics/2

Returns diagnostics and metadata associated with a mined pattern miner in a representation-independent way.

Compilation flags:
```
static
```

Template:
```
diagnostics(PatternMiner,Diagnostics)
```
Mode and number of proofs:
```
diagnostics(+compound,-list(compound)) - one
```

---

diagnostic/2

Tests or enumerates individual diagnostics metadata terms for a mined pattern miner.

Compilation flags:
```
static
```

Template:
```
diagnostic(PatternMiner,Diagnostic)
```
Mode and number of proofs:
```
diagnostic(+compound,?compound) - zero_or_more
```

---

pattern_miner_options/2

Returns the effective mining options recorded in a mined pattern miner diagnostics list.

Compilation flags:
`static`

Template:
`pattern_miner_options(PatternMiner,Options)`
Mode and number of proofs:
`pattern_miner_options(+compound,-list(compound))` - `zero_or_one`

---

check_pattern_miner/1

Checks that the argument is a structurally valid mined pattern miner term for the receiving pattern miner implementation, throwing an exception on invalid input when applicable.

Compilation flags:
`static`

Template:
`check_pattern_miner(PatternMiner)`
Mode and number of proofs:
`check_pattern_miner(@compound)` - `one_or_error`

Exceptions:
`PatternMiner` is a variable:
`instantiation_error`
`PatternMiner` is neither a variable nor a valid pattern miner:
`domain_error(pattern_miner,PatternMiner)`

---

valid_pattern_miner/1

True when `check_pattern_miner/1` succeeds for the argument without throwing an exception.

Compilation flags:
`static`

Template:
`valid_pattern_miner(PatternMiner)`
Mode and number of proofs:
`valid_pattern_miner(@compound)` - `zero_or_one`

---

export_to_clauses/4

Converts a pattern miner result into a list of predicate clauses. `Functor` is the functor for the generated predicate clauses.

Compilation flags:
`static`

Template:
`export_to_clauses(Dataset,PatternMiner,Functor,Clauses)`
Mode and number of proofs:
`export_to_clauses(+object_identifier,+compound,+callable,-list(clause))` - `one`

---

export_to_file/4

Exports a pattern miner result to a file. `Functor` is the functor for the generated predicate clauses.

Compilation flags:
`static`

Template:
`export_to_file(Dataset,PatternMiner,Functor,File)`
Mode and number of proofs:
`export_to_file(+object_identifier,+compound,+callable,+atom)` - `one`

---

print_pattern_miner/1

Prints a pattern miner result to the current output stream in a human-readable format.

Compilation flags:
`static`

Template:
`print_pattern_miner(PatternMiner)`
Mode and number of proofs:
`print_pattern_miner(+compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

transaction_dataset_protocol, *sequence_dataset_protocol*

1.254 pca_projection

object

1.254.1 `pca_projection`

Principal Component Analysis reducer for continuous datasets using a portable power-iteration eigensolver.

Availability:
`logtalk_load(pca_projection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *dimension_reducer_common*

Uses:
format
linear_algebra
list
pairs
type

Remarks:
(none)

Inherited public predicates:
check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

lda_projection, *random_projection*

1.255 pddl_parser

object

1.255.1 `pddl`

Simple parser of PDDL 3.0 files.

Availability:
`logtalk_load(pddl_parser(loader))`

Author: Robert Sasak, Charles University in Prague. Adapted to Logtalk by Paulo Moura.
Version: 1:2:2
Date: 2024-03-14

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *read_file*
Uses:
user

Remarks:
(none)

Inherited public predicates:
read_file/2

- *Public predicates*
  - *parse_domain/3*
  - *parse_domain/2*
  - *parse_problem/2*
  - *parse_problem/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse_domain/3

Parses a PDDL 3.0 domain file, returning a compound term representing its contents and rest of the file. Useful when domain and problem are in one file.

Compilation flags:
`static`

Template:
`parse_domain(File,Output,RestOfFile)`
Mode and number of proofs:
`parse_domain(+atom,-compound,-list(atom))` - `one`

---

parse_domain/2

Parses a PDDL 3.0 domain file, returning a compound term representing its contents.

Compilation flags:
`static`

Template:
`parse_domain(File,Output)`
Mode and number of proofs:
`parse_domain(+atom,-compound)` - `one`

---

parse_problem/2

Parses a PDDL 3.0 problem file, returning a compound term representing its contents.

Compilation flags:
```
static
```

Template:
```
parse_problem(File,Output)
```
Mode and number of proofs:
```
parse_problem(+atom,-compound) - one
```

---

parse_problem/3

Parses a PDDL 3.0 problem file, returning a compound term representing its contents and rest of the file. Useful when domain and problem are in one file.

Compilation flags:
```
static
```

Template:
```
parse_problem(File,Output,RestOfFile)
```
Mode and number of proofs:
```
parse_problem(+atom,-compound,-list(atom)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

category

1.255.2 read_file

Utility predicates for parsing a file as a list of atoms.

Availability:
```
logtalk_load(pddl_parser(loader))
```

Author: Robert Sasak, Charles University in Prague. Adapted to Logtalk by Paulo Moura.

Version: 1:0:0
Date: 2011-08-04

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *read_file/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

read_file/2

Reads a file character by character, parsing it into a list of atoms.

Compilation flags:
`static`

Template:
`read_file(File,List)`
Mode and number of proofs:
`read_file(+atom,-list(atom))-one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.256 permutations

object

1.256.1 `permutations`

Implementation of permutations operations over lists.

Availability:
`logtalk_load(permutations(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *permutations_protocol*
Uses:
fast_random(Algorithm)
list
natural

Remarks:
(none)

Inherited public predicates:
count_distinct_permutations/2 *count_permutations/2* *distinct_permutation/2* *distinct_permutation/3* *distinct_permutation_index/3* *distinct_permutations/2* *distinct_permutations/3* *k_permutation/3* *k_permutation/4* *k_permutations/3* *k_permutations/4* *next_permutation/2* *nth_distinct_permutation/3* *nth_permutation/3* *nth_permutation/4* *permutation/2* *permutation/3* *permutation_index/3* *permutation_index/4* *permutations/2* *permutations/3* *previous_permutation/2* *random_distinct_permutation/2* *random_permutation/2* *sample_distinct_permutations/3* *sample_permutations/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.256.2 `permutations_protocol`

Protocol for permutations operations over lists.

Availability:
`logtalk_load(permutations(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-05-12

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *permutations/2*
  - *permutation/2*
  - *permutations/3*
  - *permutation/3*
  - *distinct_permutations/2*
  - *distinct_permutation/2*
  - *distinct_permutations/3*
  - *distinct_permutation/3*

      - *k_permutations/3*
      - *k_permutation/3*
      - *k_permutations/4*
      - *k_permutation/4*
      - *next_permutation/2*
      - *previous_permutation/2*
      - *nth_permutation/3*
      - *nth_permutation/4*
      - *permutation_index/3*
      - *permutation_index/4*
      - *count_permutations/2*
      - *count_distinct_permutations/2*
      - *nth_distinct_permutation/3*
      - *distinct_permutation_index/3*
      - *random_permutation/2*
      - *sample_permutations/3*
      - *random_distinct_permutation/2*
      - *sample_distinct_permutations/3*
    - *Protected predicates*
    - *Private predicates*
    - *Operators*

Public predicates

permutations/2

Generates all permutations of a list using default order.

Compilation flags:
    `static`

Template:
    `permutations(List,Permutations)`
Mode and number of proofs:
    `permutations(+list,-list) - one`

---

permutation/2

True iff the second argument is a permutation of the first argument using default order.

Compilation flags:
    `static`

Template:
    `permutation(List,Permutation)`
Mode and number of proofs:

`permutation(+list,-list)` - `one_or_more`

---

permutations/3

Generates all permutations with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`permutations(List,Order,Permutations)`
Mode and number of proofs:
`permutations(+list,+atom,-list)` - `one`

---

permutation/3

True iff the third argument is a permutation with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`permutation(List,Order,Permutation)`
Mode and number of proofs:
`permutation(+list,+atom,-list)` - `one_or_more`

---

distinct_permutations/2

Generates all distinct permutations of a list (deduplicating repeated values in the input list) using default order.

Compilation flags:
`static`

Template:
`distinct_permutations(List,Permutations)`
Mode and number of proofs:
`distinct_permutations(+list,-list)` - `one`

---

distinct_permutation/2

True iff the second argument is a distinct permutation of the first argument using default order.

Compilation flags:
`static`

Template:
`distinct_permutation(List,Permutation)`
Mode and number of proofs:
`distinct_permutation(+list,-list)` - `one_or_more`

---

distinct_permutations/3

Generates all distinct permutations with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`distinct_permutations(List,Order,Permutations)`
Mode and number of proofs:
`distinct_permutations(+list,+atom,-list)` - `one`

---

distinct_permutation/3

True iff the third argument is a distinct permutation with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`distinct_permutation(List,Order,Permutation)`
Mode and number of proofs:
`distinct_permutation(+list,+atom,-list)` - `one_or_more`

---

k_permutations/3

Generates all K-permutations (ordered selections) of a list using default order.

Compilation flags:
`static`

Template:
`k_permutations(K,List,Permutations)`
Mode and number of proofs:
`k_permutations(+integer,+list,-list)` - `one`

---

k_permutation/3

True iff the third argument is a K-permutation (ordered selection) of a list using default order.

Compilation flags:
`static`

Template:
`k_permutation(K,List,Permutation)`
Mode and number of proofs:
`k_permutation(+integer,+list,-list)` - `one_or_more`

---

k_permutations/4

Generates all K-permutations with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`k_permutations(K,List,Order,Permutations)`
Mode and number of proofs:
`k_permutations(+integer,+list,+atom,-list)` - `one`

---

k_permutation/4

True iff the fourth argument is a K-permutation with the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`k_permutation(K,List,Order,Permutation)`
Mode and number of proofs:
`k_permutation(+integer,+list,+atom,-list)` - `one_or_more`

---

next_permutation/2

Returns the next permutation value in lexicographic order.

Compilation flags:
`static`

Template:
`next_permutation(Permutation,Next)`
Mode and number of proofs:
`next_permutation(+list,-list)` - `zero_or_one`

---

previous_permutation/2

Returns the previous permutation value in lexicographic order.

Compilation flags:
`static`

Template:
`previous_permutation(Permutation,Previous)`
Mode and number of proofs:
`previous_permutation(+list,-list)` - `zero_or_one`

---

nth_permutation/3

Returns the permutation at a given zero-based index using default order.

Compilation flags:
`static`

Template:
`nth_permutation(List,Index,Permutation)`
Mode and number of proofs:
`nth_permutation(+list,+integer,-list)` - `zero_or_one`

---

nth_permutation/4

Returns the permutation at a given zero-based index in the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`nth_permutation(List,Order,Index,Permutation)`
Mode and number of proofs:
`nth_permutation(+list,+atom,+integer,-list)` - `zero_or_one`

---

permutation_index/3

Returns the zero-based index of a permutation using default order.

Compilation flags:
`static`

Template:
`permutation_index(List,Permutation,Index)`
Mode and number of proofs:
`permutation_index(+list,+list,-integer)` - `zero_or_one`

---

permutation_index/4

Returns the zero-based index of a permutation in the given order: `default`, `lexicographic`, or `shortlex`.

Compilation flags:
`static`

Template:
`permutation_index(List,Order,Permutation,Index)`
Mode and number of proofs:
`permutation_index(+list,+atom,+list,-integer)` - `zero_or_one`

---

count_permutations/2

Counts the number of permutations of a list.

Compilation flags:
`static`

Template:
`count_permutations(List,Count)`
Mode and number of proofs:
`count_permutations(+list,-integer)` - `one`

---

count_distinct_permutations/2

Counts the number of distinct permutations of a list (deduplicating equal-valued permutations).

Compilation flags:
`static`

Template:
`count_distinct_permutations(List,Count)`
Mode and number of proofs:
`count_distinct_permutations(+list,-integer)` - `one`

---

nth_distinct_permutation/3

Returns the distinct permutation at a given zero-based index in default generation order.

Compilation flags:
`static`

Template:
`nth_distinct_permutation(List,Index,Permutation)`
Mode and number of proofs:
`nth_distinct_permutation(+list,+integer,-list)` - `zero_or_one`

---

distinct_permutation_index/3

Returns the zero-based index of a distinct permutation in default generation order.

Compilation flags:
`static`

Template:
`distinct_permutation_index(List,Permutation,Index)`
Mode and number of proofs:
`distinct_permutation_index(+list,+list,-integer)` - `zero_or_one`

---

random_permutation/2

Returns a random permutation of a list.

Compilation flags:
`static`

Template:
`random_permutation(List,Permutation)`
Mode and number of proofs:
`random_permutation(+list,-list)` - `one`

---

sample_permutations/3

Returns SampleCount random permutations of a list, sampled with replacement.

Compilation flags:
```
static
```

Template:
```
sample_permutations(List,SampleCount,Samples)
```
Mode and number of proofs:
```
sample_permutations(+list,+integer,-list) - zero_or_one
```

---

random_distinct_permutation/2

Returns a random distinct permutation of a list (deduplicating equal-valued permutations).

Compilation flags:
```
static
```

Template:
```
random_distinct_permutation(List,Permutation)
```
Mode and number of proofs:
```
random_distinct_permutation(+list,-list) - one
```

---

sample_distinct_permutations/3

Returns SampleCount random distinct permutations of a list, sampled with replacement after deduplicating equal-valued permutations.

Compilation flags:
```
static
```

Template:
```
sample_distinct_permutations(List,SampleCount,Samples)
```
Mode and number of proofs:
```
sample_distinct_permutations(+list,+integer,-list) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.257 plackett_luce_last_ranker

object

1.257.1 `plackett_luce_last_ranker`

Tie-aware Plackett-Luce-last grouped-ranking ranker. Learns one positive preference strength parameter per item from a dataset object implementing the `ranking_dataset_protocol` protocol by fitting a last-choice Plackett-Luce model to grouped tie blocks, and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:
`logtalk_load(plackett_luce_last_ranker(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ranking_dataset_common*
`public` *grouped_strength_ranker_common*
`public` *score_ranker_common*
`public` *plackett_luce_common*

Uses:
avltree
format
list
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2*

multiplayer_dataset_items/2 *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2*
multiplayer_dataset_summary/2 *option/2* *option/3* *pairwise_dataset_connected_components/2*
pairwise_dataset_declared_items/2 *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2*
pairwise_dataset_summary/2 *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2*
pairwise_measurement_dataset_declared_items/2 *pairwise_measurement_dataset_items/2*
pairwise_measurement_dataset_measurements/2 *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3*
ranker_options/2 *scores/2* *temporal_pairwise_dataset_connected_components/2*
temporal_pairwise_dataset_declared_items/2 *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3*
temporal_pairwise_dataset_items/2 *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2*
valid_option/1 *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2*
validate_multiplayer_dataset/1 *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2*
validate_pairwise_measurement_dataset/1 *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1*
validate_temporal_pairwise_dataset/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ranking_dataset_protocol, *ranker_protocol*, *borda_ranker*

1.258 plackett_luce_ranker

object

1.258.1 plackett_luce_ranker

Tie-aware Plackett-Luce grouped-ranking ranker. Learns one positive preference strength parameter per item from a dataset object implementing the `ranking_dataset_protocol` protocol by fitting a top-choice Plackett-Luce model to grouped tie blocks, and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:

```
logtalk_load(plackett_luce_ranker(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *ranking_dataset_common*
`public` *grouped_strength_ranker_common*
`public` *score_ranker_common*
`public` *plackett_luce_common*

Uses:

avltree
format
list
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ranking_dataset_protocol, *ranker_protocol*, *plackett_luce_last_ranker*, *borda_ranker*

1.259 pls_projection

object

1.259.1 `pls_projection`

Partial Least Squares projection for target-valued continuous datasets using deterministic PLS1 deflation.

Availability:
`logtalk_load(pls_projection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *dimension_reducer_common*
Uses:
format
linear_algebra
list

numberlist
pairs
population
type

Remarks:
(none)

Inherited public predicates:
check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

lda_projection, *pca_projection*

1.260 ports_profiler

object

1.260.1 ports_profiler

Predicate execution box model port profiler.

Availability:
`logtalk_load(ports_profiler(loader))`

Author: Paulo Moura
Version: 3:1:0
Date: 2026-10-30

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *tool_diagnostics_common*
`public` *options*

Provides:
logtalk::debug_handler/1
logtalk::debug_handler/3

Uses:
list
logtalk
os
type
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostic/3* *diagnostic_rule/5* *diagnostic_rules/1* *diagnostic_target/1* *diagnostics/2* *diagnostics/3* *diagnostics_preflight/2* *diagnostics_preflight/3* *diagnostics_summary/2* *diagnostics_summary/3* *diagnostics_tool/5* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *start/0*
  - *stop/0*
  - *data/0*
  - *data/1*
  - *data/2*
  - *reset/0*
  - *reset/1*
  - *port/5*
  - *clause_location/6*
  - *clause/5*
- *Protected predicates*
- *Private predicates*

- *clause_location_/6*
- *port_/5*
- *clause_/5*
- *entity_defines_/2*

- *Operators*

Public predicates

start/0

Activates the ports profiler for followup goals.

Compilation flags:
`static`

Mode and number of proofs:
`start` - `one`

---

stop/0

Deactivates the ports profiler.

Compilation flags:
`static`

Mode and number of proofs:
`stop` - `one`

---

data/0

Prints a table with all port profiling data.

Compilation flags:
`static`

Mode and number of proofs:
`data` - `one`

---

data/1

Prints a table with all port profiling data for the specified entity.

Compilation flags:

```
static
```

Template:

```
data(Entity)
```

Mode and number of proofs:

```
data(+entity_identifier) - one
```

---

data/2

Prints a table with all port profiling data for the specified entity predicate (or non-terminal).

Compilation flags:

```
static
```

Template:

```
data(Entity,Predicate)
```

Mode and number of proofs:

```
data(+entity_identifier,+predicate_indicator) - one
data(+entity_identifier,+non_terminal_indicator) - one
```

---

reset/0

Resets all port profiling data.

Compilation flags:

```
static
```

Mode and number of proofs:

```
reset - one
```

---

reset/1

Resets all port profiling data for the specified entity.

Compilation flags:

```
static
```

Template:

```
reset(Entity)
```

Mode and number of proofs:

```
reset(+entity_identifier) - one
```

---

port/5

Enumerates, by backtracking, all collected port profiling data.

Compilation flags:

```
static
```

Template:

```
port(Port,Entity,Functor,Arity,Count)
```

Mode and number of proofs:

```
port(?atom,?entity_identifier,?atom,?integer,?integer) - zero_or_more
```

---

clause_location/6

Enumerates, by backtracking, all collected profiled clause location data.

Compilation flags:

```
static
```

Template:

```
clause_location(Entity,Functor,Arity,ClauseNumber,File,BeginLine)
```

Mode and number of proofs:

```
clause_location(?entity_identifier,?atom,?integer,?integer,?atom,?integer) -
zero_or_more
```

---

clause/5

Enumerates, by backtracking, all collected clause profiling data.

Compilation flags:

```
dynamic
```

Template:

```
clause(Entity,Functor,Arity,ClauseNumber,Count)
```

Mode and number of proofs:

```
clause(?entity_identifier,?atom,?integer,?integer,?integer) - zero_or_more
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

clause_location_/6

Internal table of collected profiled clause location data.

Compilation flags:
`dynamic`

Template:
`clause_location_(Entity,Functor,Arity,ClauseNumber,File,BeginLine)`
Mode and number of proofs:
`clause_location_(?entity_identifier,?atom,?integer,?integer,?atom,?integer)` - `zero_or_more`

---

port_/5

Internal table of collected port profiling data.

Compilation flags:
`dynamic`

Template:
`port_(Port,Entity,Functor,Arity,Count)`
Mode and number of proofs:
`port_(?atom,?entity_identifier,?atom,?integer,?integer)` - `zero_or_more`

---

clause_/5

Internal table of collected clause profiling data.

Compilation flags:
`dynamic`

Template:
`clause_(Entity,Functor,Arity,ClauseNumber,Count)`
Mode and number of proofs:
`clause_(?entity_identifier,?atom,?integer,?integer,?integer)` - `zero_or_more`

---

entity_defines_/2

Internal cache for profiled predicates.

Compilation flags:
`dynamic`

Template:
`entity_defines_(Entity,Predicate)`
Mode and number of proofs:
`entity_defines_(?entity_identifier,?predicate_indicator)` - `zero_or_more`

Operators

(none)

1.261 prefix_span_pattern_miner

object

1.261.1 prefix_span_pattern_miner

PrefixSpan sequential pattern miner for sequence datasets using recursive projected databases.

Availability:
`logtalk_load(prefix_span_pattern_miner(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *sequential_pattern_mining_common*
Uses:
avltree
format
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_pattern_miner/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *mine/2* *mine/3* *option/2* *option/3* *pattern_miner_options/2* *print_pattern_miner/1* *valid_option/1* *valid_options/1* *valid_pattern_miner/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pattern_miner_protocol, *sequence_dataset_protocol*

1.262 probabilistic_pca_projection

object

1.262.1 `probabilistic_pca_projection`

Probabilistic Principal Component Analysis reducer for continuous datasets using a portable covariance eigensolver.

Availability:
`logtalk_load(probabilistic_pca_projection(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *dimension_reducer_common*
Uses:
format
linear_algebra

list
numberlist
pairs
type

Remarks:
(none)

Inherited public predicates:
check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

kernel_pca_projection, *pca_projection*, *truncated_svd_projection*

1.263 problog

object

1.263.1 problog

ProbLog CLI exact inference, source generation, and collected result inspection.

Availability:
`logtalk_load(problog(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *problog_protocol*
Imports:
`public` *options*
Uses:
list
os
problog_output
problog_process
problog_program
reader
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *print_program/1* *print_result/1* *probability/3* *program/1* *program/2* *result_property/2* *solve/2* *solve/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.263.2 `problog_expansion`

Opt-in expansion of native ProbLog facts, rules, queries, evidence, and directives.

Availability:
`logtalk_load(problog(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
list
problog_program
type

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.263.3 `problog_output`

Strict data-only decoding of ProbLog –format prolog query probabilities.

Availability:
`logtalk_load(problog(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-12

Compilation flags:
`static, context_switching_calls`

Uses:
list
term_io
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *decode/6*
  - *valid_result/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

decode/6

Decodes a complete output document and process metadata. Nonzero status and malformed output raise problog_solver_error/4 retaining both output streams.

Compilation flags:

```
static
```

Template:

```
decode(Codes,Diagnostics,RawStatus,NormalizedStatus,SourceMap,Result)
```

Mode and number of proofs:

```
decode(+list(integer),+list(integer),+term,+compound,+list(compound),-compound) -
one_or_error
```

---

valid_result/1

True for a ground, structurally valid collected inference result.

Compilation flags:

```
static
```

Template:

```
valid_result(Result)
```

Mode and number of proofs:

```
valid_result(@term) - zero_or_one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.263.4 problog_problem_protocol

Protocol for ordered, ground ProbLog statement providers.

Availability:
`logtalk_load(problog(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-12

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *problog_statement/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

problog_statement/2

Enumerates canonical ProbLog statements in source order together with their source origins.

Compilation flags:
`static`

Template:
`problog_statement(Statement,Origin)`
Mode and number of proofs:
`problog_statement(-compound,-term)` - `zero_or_more`

Remarks:

- Statements: Statements are fully ground and use the vocabulary documented by the `problog_program` object.
- Origin: Either `unknown` or `source(File, Start, End)` where positions are positive character offsets.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.263.5 problog_process

Isolated operating-system shell transport for the ProbLog CLI.

Availability:
`logtalk_load(problog(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-12

Compilation flags:
`static, context_switching_calls`

Uses:
list
os
process
reader
type
utf_8

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *quote/3*
  - *run/7*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

quote/3

Quotes an atom as one POSIX shell or Windows PowerShell literal, without interpolation.

Compilation flags:

`static`

Template:

`quote(Platform,Argument,Quoted)`

Mode and number of proofs:

`quote(+atom,+atom,-atom)` - `one_or_error`

Exceptions:

`Platform` is a variable:

`instantiation_error`

`Platform` is neither a variable nor an atom:

`type_error(atom,Platform)`

`Platform` is an atom but neither `posix` nor `windows`:

`domain_error(problog_shell_platform,Platform)`

`Argument` is a variable:

`instantiation_error`

`Argument` is neither a variable nor an atom:

`type_error(atom,Argument)`

`Argument` is an atom but contains a null character:

`domain_error(problog_shell_argument,Argument)`

---

run/7

Runs the ProbLog CLI with an owned temporary source and separate UTF-8 output captures, returning only after cleanup.

Compilation flags:

`static`

Template:

`run(Executable,Arguments,ProgramSource,Codes,Diagnostics,RawStatus,NormalizedStatus)`

Mode and number of proofs:

`run(+atom,+list(atom),+compound,-list(code),-list(code),-term,-compound)` - `one_or_error`

Exceptions:

`Executable` is a variable:

`instantiation_error`

`Executable` is neither a variable nor an atom:

`type_error(atom,Executable)`

`Arguments` is a partial list or a list with an element `Argument` which is a variable:

`instantiation_error`

`Arguments` is neither a variable nor a list:

`type_error(list(atom),Arguments)`

An element `Argument` of the list `Arguments` is neither a variable nor an atom:

`type_error(atom,Argument)`

`Executable` or an element `Argument` of the list `Arguments` contains a null character:

`domain_error(problog_shell_argument,Argument)`

`ProgramSource` is not a ground term:

`instantiation_error`

`ProgramSource` is neither `file(Path)` nor `codes(Codes)`:

`domain_error(problog_process_source,ProgramSource)`

A launcher dependency does not exist:

`existence_error(command,Command)`

A source or capture file does not exist:

`existence_error(source_sink,Path)`

A source or capture file cannot be opened:

`permission_error(open,source_sink,Path)`

An owned directory cannot be found during cleanup:

`existence_error(directory,Directory)`

An owned file or directory cannot be removed:

`permission_error(delete,source_sink,Path)`

A character sequence is not valid UTF-8:

`representation_error(utf_8)`

A launcher operation fails:

`problog_transport(Stage,Status,Details)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.263.6 `problog_program`

Canonical ground ProbLog statement validation and source generation.

Availability:

`logtalk_load(problog(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-11-15

Compilation flags:
`static, context_switching_calls`

Uses:
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *serialize/2*
  - *serialize/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

serialize/2

Serializes a list of fully ground canonical ProbLog statements to character codes.

Compilation flags:
`static`

Template:
`serialize(Statements,Codes)`

Mode and number of proofs:
`serialize(+list,-list(integer))` - `one_or_error`

Exceptions:
`Statements` is a partial list or contains a variable:
`instantiation_error`
`Statements` is not a list:
`type_error(list,Statements)`
A statement or one of its terms is outside the documented ProbLog AST:
`domain_error(problog_statement,Statement)`

Remarks:

- Statements: The statement constructors are `fact(Head)`, `rule(Head, Body)`, `probabilistic_fact(P, Head)`, `probabilistic_rule(P, Head, Body)`, `annotated_disjunction(Alternatives, Body)`, `query(Goal)`, `evidence(Goal, Boolean)`, `directive(Goal)`, `comment(Codes)`, and `raw(Codes)`.

- Heads and positive goals: Use `function(Name, Arguments)`; a zero-arity predicate has an empty argument list. Terms are `atom(Atom)`, `number(Number)`, `variable(Name)`, `anonymous`, nested `function(Name, Arguments)`, proper `list(Elements)`, improper `list(Elements, Tail)`, and `arithmetic(Operator, Arguments)`. Logic variables are represented by ground `variable(Name)` or `anonymous` nodes, preserving variable sharing without putting Prolog variables in the AST.
- Goals: Use `true`, `function(Name, Arguments)`, `conjunction(Goals)`, `disjunction(Goals)`, `negation(Goal)`, or `comparison(Operator, Left, Right)`. Arithmetic operators are `+`, `-`, `*`, `/`, `//`, `mod`, and `**`; comparison operators are `=`, `==`, `\=`, `\==`, `<`, `=<`, `>`, and `>=`.
- Queries and evidence: `query(Goal)` writes a native query directive; `evidence(Goal, true)` and `evidence(Goal, false)` write positive and negative evidence directives.
- Annotated disjunctions: `annotated_disjunction(Alternatives, Body)` uses at least two `Probability-Head` list elements and writes `Probability::Head; ...` optionally followed by `:- Body`.
- Raw source and comments: `raw(Codes)` passes trusted native source through, adding a final newline when needed. `comment(Codes)` emits comment lines. Both use ground character-code lists.
- Probabilities: Probabilities are numbers in the inclusive range [0,1]. Annotated-disjunction probabilities must sum to at most one; a residual probability is allowed.

`serialize/4`

Serializes statements with origins and returns generated `line(Start, End, Origin)` source-map entries.

Compilation flags:

`static`

Template:

`serialize(Statements,Origins,Codes,SourceMap)`

Mode and number of proofs:

`serialize(+list,+list,-list(integer),-list(compound))` - `one_or_error`

Exceptions:

`Statements` or `Origins` is a partial list or contains a variable:

`instantiation_error`

`Statements` or `Origins` is not a list:

`type_error(list,Statements)`

An invalid statement or origin is supplied, or the list lengths differ:

`domain_error(problog_origins,Origins)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.263.7 problog_protocol

Process-backed ProbLog exact inference, source generation, and collected query probabilities.

Availability:
`logtalk_load(problog(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-14

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *solve/2*
  - *solve/3*
  - *program/1*
  - *program/2*
  - *print_program/1*
  - *probability/3*
  - *result_property/2*
  - *print_result/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

solve/2

Runs exact inference with default options. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`. Returns a ground collected result after cleanup.

Compilation flags:
`static`

Template:
`solve(Source,Result)`
Mode and number of proofs:
`solve(+compound,-compound)` - `one_or_error`

Exceptions:
`Source` is not ground:
`instantiation_error`
`Source` is ground but invalid:
`domain_error(problog_source,Source)`
An executable cannot be resolved:
`existence_error(command,Command)`
A transport operation fails:
`problog_transport(Stage,Status,Details)`
Inference fails or produces invalid output:
`problog_solver_error(Formal,Status,Diagnostics,SourceMap)`

---

solve/3

Runs exact inference with the given options options. Repeated options use the first occurrence.

Compilation flags:
`static`

Template:
`solve(Source,Result,Options)`
Mode and number of proofs:
`solve(+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Source` is not ground:
`instantiation_error`
`Source` is ground but invalid:
`domain_error(problog_source,Source)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
An executable cannot be resolved:
`existence_error(command,Command)`
A transport operation fails:
`problog_transport(Stage,Status,Details)`
Inference fails or produces invalid output:
`problog_solver_error(Formal,Status,Diagnostics,SourceMap)`

---

program/1

Prints source to the current output without invoking ProbLog.

Compilation flags:
`static`

Template:
`program(Source)`
Mode and number of proofs:
`program(+compound)` - `one_or_error`

Exceptions:
`Source` is not ground:
`instantiation_error`
`Source` is ground but invalid:
`domain_error(problog_source,Source)`

---

program/2

Returns source character codes without invoking ProbLog. Native file and codes sources are not parsed.

Compilation flags:
`static`

Template:
`program(Source,Codes)`
Mode and number of proofs:
`program(+compound,-list(integer))` - `one_or_error`

Exceptions:

`Source` is not ground:
`instantiation_error`
`Source` is ground but invalid:
`domain_error(problog_source,Source)`

`print_program/1`

Prints character codes to the current output without adding a newline.

Compilation flags:
`static`

Template:
`print_program(Codes)`
Mode and number of proofs:
`print_program(+list(integer))` - `one_or_error`

Exceptions:
`Codes` is a partial list or a list with an element `Element` which is a variable:
`instantiation_error`
`Codes` is is neither a partial list nor a list:
`type_error(codes,Codes)`
An element `Element` of the `Codes` list is neither a variable nor an integer:
`type_error(integer,Element)`
An element `Element` of the `Codes` list is an integer but not a valid character code:
`domain_error(character_code,Element)`

`probability/3`

Enumerates ground query terms and numeric probabilities in a collected result. Query order follows the CLI, not a sorting contract.

Compilation flags:
`static`

Template:
`probability(Result,Query,Probability)`
Mode and number of proofs:
`probability(+compound,?callable,?number)` - `zero_or_more`

Exceptions:
`Result` is not ground:
`instantiation_error`
`Result` is ground but invalid:
`domain_error(problog_result,Result)`

---

result_property/2

Enumerates raw_status/1, process_status/1, and diagnostics/1 properties. Diagnostics are captured stderr character codes.

Compilation flags:
static

Template:
result_property(Result,Property)
Mode and number of proofs:
result_property(+compound,?compound) - zero_or_more

---

print_result/1

Prints the collected query probabilities without invoking ProbLog.

Compilation flags:
static

Template:
print_result(Result)
Mode and number of proofs:
print_result(+compound) - one_or_error

Exceptions:
Result is not ground:
instantiation_error
Result is ground but invalid:
domain_error(problog_result,Result)

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.264 process

object

1.264.1 `process`

Portable process handling predicates.

Availability:
`logtalk_load(process(loader))`

Author: Paulo Moura
Version: 2:4:0
Date: 2026-11-12

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Supported backend Prolog systems: ECLiPSe, GNU Prolog, SICStus Prolog, Shumway Prolog, SWI-Prolog, Trealla Prolog, and XVM.
- Option `type/1` limitation: This option is ignored on ECLiPSe, GNU Prolog, and XVM.

Inherited public predicates:
(none)

- *Public predicates*
  - *create/3*
  - *wait/2*
  - *wait/3*
  - *kill/2*
  - *kill/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

create/3

Creates a new process from the given executable and list of arguments. Supported options are `process(Pid)`, `stdin(Stream)`, `stdout(Stream)`, `stderr(Stream)`, and `type(Type)`.

Compilation flags:

`static`

Template:

`create(Executable,Arguments,Options)`

Mode and number of proofs:

`create(+atom,+list(atom),+list(compound))` - `zero_or_one`

---

wait/2

Waits for a process to terminate and retrieves its exit status. The value of `Status` depends on the backend. For a successfully terminated process, `Status` is either `0` or `exit(0)`.

Compilation flags:

`static`

Template:

`wait(Pid,Status)`

Mode and number of proofs:

`wait(+integer,-integer_or_compound)` - `one`

---

wait/3

Waits once for a process to terminate, returning the backend-specific status in `RawStatus` and its normalized form in `NormalizedStatus`.

Compilation flags:

`static`

Template:

`wait(Pid,RawStatus,NormalizedStatus)`

Mode and number of proofs:

`wait(+integer,-integer_or_compound,-compound)` - `one`

Remarks:

- Normalized status: Either `exit(Code)`, `killed(Signal)`, or `unknown(RawStatus)` when native information is insufficient.
- GNU Prolog limitation: No distinction between processes that exited on their own with non-zero status and processes that were killed.

---

kill/2

Kills the given process with the specified signal (an integer or one of the following atoms: `sighup`, `sigint`, `sigkill`, or `sigterm`).

Compilation flags:
`static`

Template:
`kill(Pid,Signal)`
Mode and number of proofs:
`kill(+integer,+atom_or_integer)` - `zero_or_one`

---

kill/1

Kills the given process using the default signal (`sigkill`).

Compilation flags:
`static`

Template:
`kill(Pid)`
Mode and number of proofs:
`kill(+integer)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

os

1.265 protobuf

object

1.265.1 `protobuf`

Google Protocol Buffers binary format parser and generator.

Availability:
`logtalk_load(protobuf(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-08-04

Compilation flags:
`static, context_switching_calls`

Uses:
byte_order
ieee_754(Precision,ByteOrder,NaNRepresentation)
json(ObjectRepresentation,PairRepresentation,StringRepresentation)
list
reader

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *parse/3*
  - *generate/3*
  - *generate/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses Protocol Buffers binary data from the given source (`bytes(List)`, `stream(Stream)`, or `file(Path)`) returning a `Schema-Data` pair. When the schema is not present in the file, `Schema` is unified with `false`.

Compilation flags:
`static`

Template:
`parse(Source,Schema-Data)`
Mode and number of proofs:
`parse(++compound,--pair)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid Protocol Buffers source term:
`domain_error(protobuf_source,Source)`
`Schema` is not a valid Protocol Buffers schema for the encoded message:
`domain_error(protobuf_schema,Schema)`

---

parse/3

Parses Protocol Buffers binary data from the given source using the provided schema, returning the decoded data.

Compilation flags:
`static`

Template:
`parse(Source,Schema,Data)`
Mode and number of proofs:
`parse(++compound,++term,--term)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid Protocol Buffers source term:
`domain_error(protobuf_source,Source)`
`Schema` is not a valid Protocol Buffers schema for the encoded message:
`domain_error(protobuf_schema,Schema)`

---

generate/3

Generates Protocol Buffers binary data to the given sink (`bytes(List)`, `stream(Stream)`, or `file(Path)`) from the given schema and data. The schema is not included in the output.

Compilation flags:

`static`

Template:

`generate(Sink,Schema,Data)`

Mode and number of proofs:

`generate(++compound,++term,++term)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid Protocol Buffers sink term:

`domain_error(protobuf_sink,Sink)`

`Schema` is not a valid Protocol Buffers schema for `Data`:

`domain_error(protobuf_schema,Schema)`

---

generate/4

Generates Protocol Buffers binary data to the given sink from the given schema and data. When `IncludeSchema` is `true`, the schema is embedded in a wrapper message.

Compilation flags:

`static`

Template:

`generate(Sink,IncludeSchema,Schema,Data)`

Mode and number of proofs:

`generate(++compound,++boolean,++term,++term)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid Protocol Buffers sink term:

`domain_error(protobuf_sink,Sink)`

`Schema` is not a valid Protocol Buffers schema for `Data`:

`domain_error(protobuf_schema,Schema)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.266 qda_classifier

object

1.266.1 `qda_classifier`

Quadratic Discriminant Analysis classifier for continuous datasets using class-specific regularized covariance models.

Availability:
`logtalk_load(qda_classifier(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *classifier_common*
Uses:
format
linear_algebra
list
type

Remarks:
(none)

Inherited public predicates:
check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *predict_scores/3*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`predict_scores/3`

Predicts class discriminant scores for a new instance.

Compilation flags:
```
static
```

Template:
```
predict_scores(Classifier,Instance,Scores)
```
Mode and number of proofs:
```
predict_scores(+compound,+list,-list) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *lda_classifier*, *nearest_centroid_classifier*

1.267 queues

object

1.267.1 `queue`

Queue predicates implemented using difference lists.

Availability:
```
logtalk_load(queues(loader))
```

Author: Paulo Moura
Version: 1:3:0

Date: 2020-12-09

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *queuep*
Extends:
`public` *compound*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 append/3 as_list/2 check/1 depth/2 empty/1 ground/1 head/2 join/3 join_all/3 jump/3 jump_all/3 jump_all_block/3 length/2 map/2 map/3 new/1 numbervars/1 numbervars/3 occurs/2 serve/3 singletons/2 subsumes/2 subterm/2 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.267.2 queuep

Queue protocol.

Availability:
`logtalk_load(queues(loader))`

Author: Paulo Moura
Version: 1:3:0

Date: 2020-12-09

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *empty/1*
  - *head/2*
  - *join/3*
  - *join_all/3*
  - *jump/3*
  - *jump_all/3*
  - *jump_all_block/3*
  - *append/3*
  - *length/2*
  - *serve/3*
  - *as_list/2*
  - *map/2*
  - *map/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

empty/1

True if the queue is empty.

Compilation flags:
`static`

Template:
`empty(Queue)`

Mode and number of proofs:
`empty(@queue)` - `zero_or_one`

head/2

Unifies `Head` with the first element of the queue.

Compilation flags:
`static`

Template:
`head(Queue,Head)`
Mode and number of proofs:
`head(+queue,?term)` - `zero_or_one`

---

join/3

Adds the new element at the end of the queue.

Compilation flags:
`static`

Template:
`join(Element,Queue,NewQueue)`
Mode and number of proofs:
`join(@term,+queue,-queue)` - `zero_or_one`

---

join_all/3

Adds the new elements at the end of the queue. The elements are added in the same order that they appear in the list.

Compilation flags:
`static`

Template:
`join_all(List,Queue,NewQueue)`
Mode and number of proofs:
`join_all(+list,+queue,-queue)` - `zero_or_one`

---

jump/3

Adds the new element at the front of the queue.

Compilation flags:
`static`

Template:
`jump(Element,Queue,NewQueue)`
Mode and number of proofs:
`jump(@term,+queue,-queue)` - `zero_or_one`

---

jump_all/3

Adds the new elements at the front of the queue. The last element in the list will be at the front of the queue.

Compilation flags:
`static`

Template:
`jump_all(Elements,Queue,NewQueue)`
Mode and number of proofs:
`jump_all(+list,+queue,-queue)` - `zero_or_one`

---

jump_all_block/3

Adds the new elements as a block at the front of the queue. The first element in the list will be at the front of the queue.

Compilation flags:
`static`

Template:
`jump_all_block(Elements,Queue,NewQueue)`
Mode and number of proofs:
`jump_all_block(+list,+queue,-queue)` - `zero_or_one`

---

append/3

Appends two queues. The new queue will have the elements of the first queue followed by the elements of the second queue.

Compilation flags:
`static`

Template:
`append(Queue1,Queue2,NewQueue)`
Mode and number of proofs:
`append(+queue,+queue,-queue)` - `one`

---

length/2

Queue length.

Compilation flags:
`static`

Template:
`length(Queue,Length)`
Mode and number of proofs:
`length(+heap,?integer)` - `zero_or_one`

---

serve/3

Removes the first element of the queue for service.

Compilation flags:
`static`

Template:
`serve(Queue,Head,NewQueue)`
Mode and number of proofs:
`serve(+queue,?term,-queue)` - `zero_or_one`

---

as_list/2

Converts a queue to a list.

Compilation flags:
```
static
```

Template:
```
as_list(Queue,List)
```
Mode and number of proofs:
```
as_list(+queue,-list) - one
```

---

map/2

Applies a closure to all elements of a queue.

Compilation flags:
```
static
```

Template:
```
map(Closure,Queue)
```
Meta-predicate template:
```
map(1,*)
```
Mode and number of proofs:
```
map(+callable,+queue) - zero_or_one
```

---

map/3

Applies a closure to all elements of a queue constructing a new queue.

Compilation flags:
```
static
```

Template:
```
map(Closure,Queue,NewQueue)
```
Meta-predicate template:
```
map(2,*,*)
```
Mode and number of proofs:
```
map(+callable,+queue,?queue) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

queue

1.268 random

object

1.268.1 `backend_random`

Random number generator predicates using the backend Prolog compiler built-in random generator.

Availability:
`logtalk_load(random(loader))`

Author: Paulo Moura
Version: 1:25:0
Date: 2026-10-28

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *pseudo_random_protocol*
`public` *sampling_protocol*
Uses:
list

Remarks:

- Implementation: The backend Prolog compiler built-in random generator is only used for the basic `random/1`, `get_seed/1`, and `set_seed/1` predicates.
- Portability: B-Prolog, CxProlog, ECLiPSe, JIProlog, Qu-Prolog, and Quintus Prolog do not provide implementations for the `get_seed/1` and `set_seed/1` predicates and calling these predicates simply succeed without performing any action.

Inherited public predicates:
bernoulli/2 beta/3 between/3 binomial/3 chi_squared/2 circular_uniform_cartesian/3 circular_uniform_polar/3 dirichlet/2 enumerate/2 exponential/2 fisher/3 gamma/3 geometric/2 get_seed/1 gumbel/3 hypergeometric/4 logistic/3 lognormal/3 logseries/2 maybe/0 maybe/1 maybe/2 maybe_call/1 maybe_call/2 member/2 normal/3 permutation/2

poisson/2 *power/2* *random/1* *random/3* *randseq/4* *randset/4* *sample/3* *sample/4* *select/3* *select/4* *sequence/4* *set/4* *set_seed/1* *standard_cauchy/3* *standard_exponential/1* *standard_gamma/2* *standard_normal/1* *standard_t/2* *swap/2* *swap_consecutive/2* *triangular/4* *uniform/1* *uniform/3* *von_mises/3* *wald/3* *weibull/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

random, *fast_random*

object

1.268.2 `fast_random`

Portable random number generator predicates. Core predicates originally written by Richard O'Keefe. Based on algorithm AS 183 from Applied Statistics.

Availability:

```
logtalk_load(random(loader))
```

Author: Paulo Moura
Version: 3:0:0
Date: 2026-01-25

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public fast_random(as183)
```

Remarks:

- Single random number generator: This object provides a faster version of the `random` library object but does not support being extended to define multiple random number generators.
- Randomness: Loading this object always initializes the random generator seed to the same value, thus providing a pseudo random number generator. The `randomize/1` predicate can be used to initialize the seed with a random value.

Inherited public predicates:

bernoulli/2 beta/3 between/3 binomial/3 chi_squared/2 circular_uniform_cartesian/3 circular_uniform_polar/3 dirichlet/2 enumerate/2 exponential/2 fisher/3 gamma/3 geometric/2 get_seed/1 gumbel/3 hypergeometric/4 logistic/3 lognormal/3 logseries/2 maybe/0 maybe/1 maybe/2 maybe_call/1 maybe_call/2 member/2 normal/3 permutation/2 poisson/2 power/2 random/1 random/3 randomize/1 randseq/4 randset/4 reset_seed/0 sample/3 sample/4 select/3 select/4 sequence/4 set/4 set_seed/1 standard_cauchy/3 standard_exponential/1 standard_gamma/2 standard_normal/1 standard_t/2 swap/2 swap_consecutive/2 triangular/4 uniform/1 uniform/3 von_mises/3 wald/3 weibull/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

fast_random, *random*, *random(Algorithm)*, *backend_random*

object

1.268.3 `fast_random(Algorithm)`

- `Algorithm` - Random number generator algorithm. One of `as183`, `splitmix64`, `xoshiro128pp`, `xoshiro128ss`, `xoshiro256pp`, `xoshiro256ss`, `well512a`.

Fast portable random number generator predicates.

Availability:

`logtalk_load(random(loader))`

Author: Paulo Moura

Version: 3:2:0
Date: 2026-10-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *pseudo_random_protocol*
`public` *sampling_protocol*

Uses:
list

Remarks:

- Single random number generator: This object provides a faster version of the `random` library object but does not support being extended to define multiple random number generators.
- Randomness: Loading this object always initializes the random generator seed to the same value, thus providing a pseudo random number generator. The `randomize/1` predicate can be used to initialize the seed with a random value.
- `as183`: Algorithm AS 183 from Applied Statistics. 32-bit PRNG with period 2^60. Not cryptographically secure.
- `xoshiro128pp`: Xoshiro128++ random number generator. 32-bit state-of-the-art PRNG with period 2^128-1. Algorithm by David Blackman and Sebastiano Vigna.
- `xoshiro128ss`: Xoshiro128** random number generator. 32-bit PRNG with period 2^128-1. Algorithm by David Blackman and Sebastiano Vigna.
- `xoshiro256pp`: Xoshiro256++ random number generator. 64-bit state-of-the-art PRNG with period 2^256-1. Algorithm by David Blackman and Sebastiano Vigna.
- `xoshiro256ss`: Xoshiro256** random number generator. 64-bit PRNG with period 2^256-1. Algorithm by David Blackman and Sebastiano Vigna.
- `splitmix64`: SplitMix64 random number generator. 64-bit PRNG primarily used for seeding other generators. Algorithm by Guy L. Steele Jr. et al.
- `well512a`: WELL512a 16x32-bit state-of-the-art PRNG with period 2^512-1. Algorithm by by François Panneton, Pierre L'Ecuyer, and Makoto Matsumoto.
- Algorithms backend compatibility: The SplitMix64, Xoshiro256++, and Xoshiro256** algorithms require support for unbound integer arithmetic.

Inherited public predicates:
bernoulli/2 beta/3 between/3 binomial/3 chi_squared/2 circular_uniform_cartesian/3 circular_uniform_polar/3 dirichlet/2 enumerate/2 exponential/2 fisher/3 gamma/3 geometric/2 get_seed/1 gumbel/3 hypergeometric/4 logistic/3 lognormal/3 logseries/2 maybe/0 maybe/1 maybe/2 maybe_call/1 maybe_call/2 member/2 normal/3 permutation/2 poisson/2 power/2 random/1 random/3 randseq/4 randset/4 sample/3 sample/4 select/3 select/4 sequence/4 set/4 set_seed/1 standard_cauchy/3 standard_exponential/1 standard_gamma/2 standard_normal/1 standard_t/2 swap/2 swap_consecutive/2 triangular/4 uniform/1 uniform/3 von_mises/3 wald/3 weibull/3

- *Public predicates*
  - *reset_seed/0*
  - *randomize/1*
- *Protected predicates*
- *Private predicates*
  - *seed_/2*
- *Operators*

Public predicates

reset_seed/0

Resets the random generator seed to its default value. Use `get_seed/1` and `set_seed/1` instead if you need reproducibility.

Compilation flags:
`static, synchronized`

Mode and number of proofs:
`reset_seed` - `one`

---

randomize/1

Randomizes the random generator using a positive integer to compute a new seed. Use of a large integer is recommended. In alternative, when using a small integer argument, discard the first dozen random values.

Compilation flags:
`static, synchronized`

Template:
`randomize(Seed)`
Mode and number of proofs:
`randomize(+positive_integer)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

seed_/2

Stores the current random generator seed values (a ground but otherwise opaque term).

Compilation flags:
`dynamic`

Template:
`seed_(Algorithm,Seed)`
Mode and number of proofs:
`seed_(+atom,-ground)` - `one`

Operators

(none)

See also

fast_random, *random(Algorithm)*, *random*, *backend_random*

protocol

1.268.4 pseudo_random_protocol

Pseudo-random number generator protocol for seed handling predicates. These predicates are declared as synchronized when the library is compiled using a backend supporting threads.

Availability:
`logtalk_load(random(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2021-02-21

Compilation flags:
`static`

Extends:
`public` *random_protocol*

Remarks:
(none)

Inherited public predicates:
between/3 *enumerate/2* *maybe/0* *maybe/1* *maybe/2* *maybe_call/1* *maybe_call/2* *member/2* *permutation/2* *random/1* *random/3* *randseq/4* *randset/4* *sample/3* *sample/4* *select/3* *select/4* *sequence/4* *set/4* *swap/2* *swap_consecutive/2*

- *Public predicates*
  - *get_seed/1*
  - *set_seed/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

get_seed/1

Gets the current random generator seed. Seed should be regarded as an opaque ground term.

Compilation flags:
`static, synchronized`

Template:
`get_seed(Seed)`
Mode and number of proofs:
`get_seed(-ground)` - `one`

---

set_seed/1

Sets the random generator seed to a given value returned by calling the `get_seed/1` predicate.

Compilation flags:
`static, synchronized`

Template:
`set_seed(Seed)`
Mode and number of proofs:
`set_seed(+ground)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

random, *backend_random*, *fast_random*

object

1.268.5 `random`

Portable random number generator predicates. Core predicates originally written by Richard O'Keefe. Based on algorithm AS 183 from Applied Statistics.

Availability:

```
logtalk_load(random(loader))
```

Author: Paulo Moura
Version: 3:0:0
Date: 2026-01-24

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *random(as183)*

Remarks:

- Multiple random number generators: To define multiple random number generators, simply extend this object. The derived objects must send to *self* the `reset_seed/0` message.
- Randomness: Loading this object always initializes the random generator seed to the same value, thus providing a pseudo random number generator. The `randomize/1` predicate can be used to initialize the seed with a random value.

Inherited public predicates:

bernoulli/2 beta/3 between/3 binomial/3 chi_squared/2 circular_uniform_cartesian/3 circular_uniform_polar/3 dirichlet/2 enumerate/2 exponential/2 fisher/3 gamma/3 geometric/2 get_seed/1 gumbel/3 hypergeometric/4 logistic/3 lognormal/3 logseries/2 maybe/0 maybe/1 maybe/2 maybe_call/1 maybe_call/2 member/2 normal/3 permutation/2 poisson/2 power/2 random/1 random/3 randomize/1 randseq/4 randset/4 reset_seed/0 sample/3 sample/4 select/3 select/4 sequence/4 set/4 set_seed/1 standard_cauchy/3 standard_exponential/1 standard_gamma/2 standard_normal/1 standard_t/2 swap/2 swap_consecutive/2 triangular/4 uniform/1 uniform/3 von_mises/3 wald/3 weibull/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

fast_random, *backend_random*

object

1.268.6 `random(Algorithm)`

- `Algorithm` - Random number generator algorithm. One of `as183`, `splitmix64`, `xoshiro128pp`, `xoshiro128ss`, `xoshiro256pp`, `xoshiro256ss`, `well512a`.

Portable random number generator predicates.

Availability:
`logtalk_load(random(loader))`

Author: Paulo Moura
Version: 3:2:0
Date: 2026-10-16

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *pseudo_random_protocol*
`public` *sampling_protocol*
Uses:
list

Remarks:

- Multiple random number generators: To define multiple random number generators, simply extend this object. The derived objects must send to *self* the `reset_seed/0` message.
- Randomness: Loading this object always initializes the random generator seed to the same value, thus providing a pseudo random number generator. The `randomize/1` predicate can be used to initialize the seed with a random value.
- `as183`: Algorithm AS 183 from Applied Statistics. 32-bit PRNG with period 2^60. Not cryptographically secure.
- `xoshiro128pp`: Xoshiro128++ random number generator. 32-bit state-of-the-art PRNG with period 2^128-1. Algorithm by David Blackman and Sebastiano Vigna.

- `xoshiro128ss`: Xoshiro128** random number generator. 32-bit PRNG with period 2^128-1. Algorithm by David Blackman and Sebastiano Vigna.
- `xoshiro256pp`: Xoshiro256++ random number generator. 64-bit state-of-the-art PRNG with period 2^256-1. Algorithm by David Blackman and Sebastiano Vigna.
- `xoshiro256ss`: Xoshiro256** random number generator. 64-bit PRNG with period 2^256-1. Algorithm by David Blackman and Sebastiano Vigna.
- `splitmix64`: SplitMix64 random number generator. 64-bit PRNG primarily used for seeding other generators. Algorithm by Guy L. Steele Jr. et al.
- `well512a`: WELL512a 16x32-bit state-of-the-art PRNG with period 2^512-1. Algorithm by by François Panneton, Pierre L'Ecuyer, and Makoto Matsumoto.
- Algorithms backend compatibility: The SplitMix64, Xoshiro256++, and Xoshiro256** algorithms require support for unbound integer arithmetic.

Inherited public predicates:

bernoulli/2 beta/3 between/3 binomial/3 chi_squared/2 circular_uniform_cartesian/3 circular_uniform_polar/3 dirichlet/2 enumerate/2 exponential/2 fisher/3 gamma/3 geometric/2 get_seed/1 gumbel/3 hypergeometric/4 logistic/3 lognormal/3 logseries/2 maybe/0 maybe/1 maybe/2 maybe_call/1 maybe_call/2 member/2 normal/3 permutation/2 poisson/2 power/2 random/1 random/3 randseq/4 randset/4 sample/3 sample/4 select/3 select/4 sequence/4 set/4 set_seed/1 standard_cauchy/3 standard_exponential/1 standard_gamma/2 standard_normal/1 standard_t/2 swap/2 swap_consecutive/2 triangular/4 uniform/1 uniform/3 von_mises/3 wald/3 weibull/3

- *Public predicates*
  - *reset_seed/0*
  - *randomize/1*
- *Protected predicates*
- *Private predicates*
  - *seed_/2*
- *Operators*

Public predicates

reset_seed/0

Resets the random generator seed to its default value. Use `get_seed/1` and `set_seed/1` instead if you need reproducibility.

Compilation flags:

`static, synchronized`

Mode and number of proofs:

`reset_seed` - `one`

randomize/1

Randomizes the random generator using a positive integer to compute a new seed. Use of a large integer is recommended. In alternative, when using a small integer argument, discard the first dozen random values.

Compilation flags:

```
static, synchronized
```

Template:

```
randomize(Seed)
```

Mode and number of proofs:

```
randomize(+positive_integer) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

seed_/2

Stores the current random generator seed (a ground but otherwise opaque term).

Compilation flags:

```
dynamic
```

Template:

```
seed_(Algorithm,Seed)
```

Mode and number of proofs:

```
seed_(+atom,-ground) - one
```

Operators

(none)

See also

random, *fast_random(Algorithm)*, *backend_random*

protocol

1.268.7 random_protocol

Random number generator protocol. The predicates are declared as synchronized when the library is compiled using a backend supporting threads.

Availability:

```
logtalk_load(random(loader))
```

Author: Paulo Moura
Version: 3:4:0
Date: 2026-10-16

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *random/1*
  - *between/3*
  - *member/2*
  - *select/3*
  - *select/4*
  - *swap/2*
  - *swap_consecutive/2*
  - *enumerate/2*
  - *sample/3*
  - *sample/4*
  - *permutation/2*
  - *sequence/4*
  - *set/4*
  - *random/3*
  - *randseq/4*
  - *randset/4*
  - *maybe/0*
  - *maybe/1*
  - *maybe/2*
  - *maybe_call/1*
  - *maybe_call/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

random/1

Returns a new random float value in the interval `[0.0, 1.0[`.

Compilation flags:
`static, synchronized`

Template:
`random(Random)`
Mode and number of proofs:
`random(-float)` - `one`

---

between/3

Returns a new random integer in the interval `[Lower, Upper]`. Fails if `Lower` or `Upper` are not integers or if `Lower > Upper`.

Compilation flags:
`static`

Template:
`between(Lower,Upper,Random)`
Mode and number of proofs:
`between(+integer,+integer,-integer)` - `zero_or_one`

---

member/2

Returns a random member of a list. Fails if the list is empty.

Compilation flags:
`static`

Template:
`member(Random,List)`
Mode and number of proofs:
`member(-term,+list(term))` - `zero_or_one`

---

select/3

Returns a random member of a list and the rest of the list. Fails if the list is empty.

Compilation flags:
`static`

Template:
`select(Random,List,Rest)`
Mode and number of proofs:
`select(-term,+list(term),-list(term))` - `zero_or_one`

---

select/4

Returns a random member of a list, replacing it with a new element and returning the resulting list.

Compilation flags:
`static`

Template:
`select(Random,OldList,New,NewList)`
Mode and number of proofs:
`select(-term,+list(term),@term,-list(term))` - `zero_or_one`

---

swap/2

Swaps two randomly selected elements of a list. Fails if the list is empty or contains a single element.

Compilation flags:
`static`

Template:
`swap(OldList,NewList)`
Mode and number of proofs:
`swap(-term,+list(term))` - `zero_or_one`

---

swap_consecutive/2

Swaps two randomly selected consecutive elements of a list. Fails if the list is empty or contains a single element.

Compilation flags:
```
static
```

Template:
```
swap_consecutive(OldList,NewList)
```
Mode and number of proofs:
```
swap_consecutive(-term,+list(term)) - zero_or_one
```

---

enumerate/2

Enumerates the elements of a list in random order. Fails if the list is empty.

Compilation flags:
```
static
```

Template:
```
enumerate(List,Random)
```
Mode and number of proofs:
```
enumerate(+list(term),--term) - zero_or_more
```

---

sample/3

Returns a random sample without replacement, preserving source order and element identity. Returns the whole list when the requested size is at least its length.

Compilation flags:
```
static
```

Template:
```
sample(List,Size,Sample)
```
Mode and number of proofs:
```
sample(+list,+non_negative_integer,-list) - one
```

---

sample/4

Returns a random sample without replacement using the supplied list length, preserving source order and element identity. The supplied length must equal the actual list length and is not verified. Returns the whole list when the requested size is at least its length.

Compilation flags:

```
static
```

Template:

```
sample(List,Length,Size,Sample)
```

Mode and number of proofs:

```
sample(+list,+non_negative_integer,+non_negative_integer,-list) - one
```

---

permutation/2

Returns a random permutation of a list.

Compilation flags:

```
static, synchronized
```

Template:

```
permutation(List,Permutation)
```

Mode and number of proofs:

```
permutation(+list,-list) - one
```

---

sequence/4

Returns list of random integers of given length in random order in interval `[Lower, Upper]`. Fails if `Length`, `Lower`, or `Upper` are not integers or if `Lower > Upper`.

Compilation flags:

```
static, synchronized
```

Template:

```
sequence(Length,Lower,Upper,List)
```

Mode and number of proofs:

```
sequence(+integer,+integer,+integer,-list(integer)) - zero_or_one
```

---

set/4

Returns ordered set of random integers of given size in interval `[Lower, Upper]`. Fails if `Length`, `Lower`, or `Upper` are not integers, if `Lower > Upper`, or if `Length > Upper - Lower + 1`.

Compilation flags:

```
static, synchronized
```

Template:

```
set(Length,Lower,Upper,Set)
```

Mode and number of proofs:

```
set(+integer,+integer,+integer,-list(integer)) - zero_or_one
```

---

random/3

Returns a new random value in the interval `[Lower, Upper[`. Fails if `Lower > Upper`. Deprecated. Use `between/3` for integers.

Compilation flags:

```
static, synchronized
```

Template:

```
random(Lower,Upper,Random)
```

Mode and number of proofs:

```
random(+integer,+integer,-integer) - zero_or_one
random(+float,+float,-float) - zero_or_one
```

---

randseq/4

Returns list of random values of given length in random order in interval `[Lower, Upper[`. Fails if `Lower > Upper` or if the arguments are neither integers or floats. Deprecated. Use `sequence/4` for integers.

Compilation flags:

```
static, synchronized
```

Template:

```
randseq(Length,Lower,Upper,List)
```

Mode and number of proofs:

```
randseq(+integer,+integer,+integer,-list(integer)) - zero_or_one
randseq(+integer,+float,+float,-list(float)) - zero_or_one
```

---

randset/4

Returns ordered set of random values of given size in interval `[Lower, Upper[`. Fails if the arguments are neither integers or floats, `Lower > Upper`, or `Length > Upper - Lower` when arguments are integers. Deprecated. Use `set/4` for integers.

Compilation flags:

```
static, synchronized
```

Template:

```
randset(Length,Lower,Upper,Set)
```

Mode and number of proofs:

```
randset(+integer,+integer,+integer,-list(integer)) - zero_or_one
randset(+integer,+float,+float,-list(float)) - zero_or_one
```

---

maybe/0

Succeeds or fails with equal probability.

Compilation flags:

```
static
```

Mode and number of proofs:

```
maybe - zero_or_one
```

---

maybe/1

Succeeds with probability `Probability` or fails with probability `1 - Probability`. Fails if `Probability` is not a float or is outside the interval `[0.0, 1.0]`.

Compilation flags:

```
static
```

Template:

```
maybe(Probability)
```

Mode and number of proofs:

```
maybe(+probability) - zero_or_one
```

---

maybe/2

Succeeds with probability `K/N` where `K` and `N` are integers satisfying the equation `0 =< K =< N`. Fails otherwise.

Compilation flags:
`static`

Template:
`maybe(K,N)`
Mode and number of proofs:
`maybe(+non_negative_integer,+non_negative_integer)` - `zero_or_one`

---

maybe_call/1

Calls a goal or fails without calling it with equal probability. When the goal is called, it determines if this predicate succeeds once or fails.

Compilation flags:
`static`

Template:
`maybe_call(Goal)`
Meta-predicate template:
`maybe_call(0)`
Mode and number of proofs:
`maybe_call(+callable)` - `zero_or_one`

---

maybe_call/2

Calls a goal or fails without calling it with probability `Probability`. When the goal is called, it determines if this predicate succeeds once or fails.

Compilation flags:
`static`

Template:
`maybe_call(Probability,Goal)`
Meta-predicate template:
`maybe_call(*,0)`
Mode and number of proofs:
`maybe_call(+probability,+callable)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

random, *backend_random*, *fast_random*

protocol

1.268.8 `sampling_protocol`

Predicates for sampling probability distributions.

Availability:
`logtalk_load(random(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-08-10

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *normal/3*
  - *lognormal/3*
  - *wald/3*
  - *chi_squared/2*
  - *fisher/3*
  - *logseries/2*
  - *geometric/2*

- hypergeometric/4
- exponential/2
- binomial/3
- bernoulli/2
- beta/3
- gamma/3
- logistic/3
- poisson/2
- power/2
- weibull/3
- uniform/3
- uniform/1
- triangular/4
- von_mises/3
- gumbel/3
- dirichlet/2
- circular_uniform_polar/3
- circular_uniform_cartesian/3
- standard_t/2
- standard_cauchy/3
- standard_exponential/1
- standard_gamma/2
- standard_normal/1

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

normal/3

Returns a scaled normally (Gaussian) distributed random value with the given mean and standard deviation.

Compilation flags:

```
static
```

Template:

```
normal(Mean,Deviation,Value)
```

Mode and number of proofs:

```
normal(+float,+non_negative_float,-float) - one
```

lognormal/3

Returns a scaled log normally distributed random value with the given mean and standard deviation for the normal distribution.

Compilation flags:

```
static
```

Template:

```
lognormal(Mean,Deviation,Value)
```

Mode and number of proofs:

```
lognormal(+float,+non_negative_float,-float) - one
```

---

wald/3

Returns a scaled Wald (inverse Gaussian) distributed random value with the given mean.

Compilation flags:

```
static
```

Template:

```
wald(Mean,Scale,Value)
```

Mode and number of proofs:

```
wald(+positive_float,+positive_float,-float) - one
```

---

chi_squared/2

Returns a chi-squared distributed random value given the degrees of freedom.

Compilation flags:

```
static
```

Template:

```
chi_squared(DegreesOfFreedom,Value)
```

Mode and number of proofs:

```
chi_squared(+positive_integer,-float) - one
```

---

fisher/3

Returns a Fisher distributed random value given the degrees of freedom in the numerator and in the denominator.

Compilation flags:
`static`

Template:
`fisher(DegreesOfFreedomNumerator,DegreesOfFreedomDenominator,Value)`
Mode and number of proofs:
`fisher(+positive_integer,+positive_integer,-float)` - `one`

---

logseries/2

Returns a logseries distributed random value. Requires `0.0 < Shape < 1` and fails otherwise.

Compilation flags:
`static`

Template:
`logseries(Shape,Value)`
Mode and number of proofs:
`logseries(+positive_float,-positive_integer)` - `zero_or_one`

---

geometric/2

Returns a geometric distributed random value (trials until the first success). Requires `0.0 < Probability =< 1.0` and fails otherwise.

Compilation flags:
`static`

Template:
`geometric(Probability,Value)`
Mode and number of proofs:
`geometric(+probability,-positive_integer)` - `zero_or_one`

---

hypergeometric/4

Returns a hypergeometric distributed random value. Requires `Successes =< Population` and `Draws =< Population` and fails otherwise.

Compilation flags:

`static`

Template:

`hypergeometric(Population,Successes,Draws,Value)`

Mode and number of proofs:

`hypergeometric(+non_negative_integer,+non_negative_integer,+non_negative_integer,-non_negative_integer)` - `zero_or_one`

---

exponential/2

Returns a scaled exponentially distributed random value.

Compilation flags:

`static`

Template:

`exponential(Scale,Value)`

Mode and number of proofs:

`exponential(+positive_float,-float)` - `one`

---

binomial/3

Returns a binomial distributed random value.

Compilation flags:

`static`

Template:

`binomial(Trials,Probability,Value)`

Mode and number of proofs:

`binomial(+non_negative_integer,+probability,-non_negative_integer)` - `one`

---

bernoulli/2

Returns a Bernoulli distributed random value.

Compilation flags:
`static`

Template:
`bernoulli(Probability,Value)`
Mode and number of proofs:
`bernoulli(+probability,-non_negative_integer)` - `one`

---

beta/3

Returns a beta distributed random value.

Compilation flags:
`static`

Template:
`beta(Alpha,Beta,Value)`
Mode and number of proofs:
`beta(+positive_float,+positive_float,-float)` - `one`

---

gamma/3

Returns a scaled gamma distributed random value.

Compilation flags:
`static`

Template:
`gamma(Shape,Scale,Value)`
Mode and number of proofs:
`gamma(+positive_float,+positive_float,-float)` - `one`

---

logistic/3

Returns a scaled logistic distributed random value.

Compilation flags:
```
static
```

Template:
```
logistic(Location,Scale,Value)
```
Mode and number of proofs:
```
logistic(+float,+positive_float,-float) - one
```

---

poisson/2

Returns a Poisson distributed random value given the expected number of events.

Compilation flags:
```
static
```

Template:
```
poisson(Mean,Value)
```
Mode and number of proofs:
```
poisson(+non_negative_float,-non_negative_integer) - one
```

---

power/2

Returns a power distributed random value.

Compilation flags:
```
static
```

Template:
```
power(Exponent,Value)
```
Mode and number of proofs:
```
power(+positive_float,-float) - one
```

---

weibull/3

Returns a scaled Weibull distributed random value.

Compilation flags:
`static`

Template:
`weibull(Shape,Scale,Value)`
Mode and number of proofs:
`weibull(+positive_float,+positive_float,-float)` - `one`

---

uniform/3

Returns a uniform distributed random value in the interval `[Lower, Upper[`. Fails if `Lower` or `Upper` are not floats or if `Lower > Upper`. Same as `random/3`.

Compilation flags:
`static`

Template:
`uniform(Lower,Upper,Value)`
Mode and number of proofs:
`uniform(+float,+float,-float)` - `zero_or_one`

---

uniform/1

Returns a uniform distributed random value in the interval `[0.0, 1.0[`. Same as `random/1`.

Compilation flags:
`static`

Template:
`uniform(Value)`
Mode and number of proofs:
`uniform(-float)` - `one`

---

triangular/4

Returns a triangular distributed random value. Returns `Left` when `Left`, `Mode`, and `Right` are equal. Fails if the `Left =< Mode =< Right` condition does not hold.

Compilation flags:
`static`

Template:
`triangular(Left,Mode,Right,Value)`
Mode and number of proofs:
`triangular(+float,+float,+float,-float)` - `zero_or_one`

---

von_mises/3

Returns a von Mises distributed random value.

Compilation flags:
`static`

Template:
`von_mises(Mode,Concentration,Value)`
Mode and number of proofs:
`von_mises(+float,+non_negative_float,-float)` - `one`

---

gumbel/3

Returns a Gumbel distributed random value. Fails if `Scale` is not positive.

Compilation flags:
`static`

Template:
`gumbel(Location,Scale,Value)`
Mode and number of proofs:
`gumbel(+float,+positive_float,-float)` - `one`

---

dirichlet/2

Returns a Dirichlet distributed list of random values.

Compilation flags:
`static`

Template:
`dirichlet(Alphas,Thetas)`
Mode and number of proofs:
`dirichlet(+list(positive_float),-list(positive_float))` - `one`

---

circular_uniform_polar/3

Returns a circular uniform distributed random point in polar coordinates given the circle radius.

Compilation flags:
`static`

Template:
`circular_uniform_polar(Radius,Rho,Theta)`
Mode and number of proofs:
`circular_uniform_polar(+non_negative_float,-float,-float)` - `one`

---

circular_uniform_cartesian/3

Returns a circular uniform distributed random point in cartesian coordinates given the circle radius.

Compilation flags:
`static`

Template:
`circular_uniform_cartesian(Radius,X,Y)`
Mode and number of proofs:
`circular_uniform_cartesian(+non_negative_float,-float,-float)` - `one`

---

standard_t/2

Returns a standard Student's t distributed random value given the degrees of freedom.

Compilation flags:
```
static
```

Template:
```
standard_t(DegreesOfFreedom,Value)
```
Mode and number of proofs:
```
standard_t(+positive_integer,-float) - one
```

---

standard_cauchy/3

Returns a standard Cauchy distributed random value.

Compilation flags:
```
static
```

Template:
```
standard_cauchy(Location,Scale,Value)
```
Mode and number of proofs:
```
standard_cauchy(+float,+positive_float,-float) - one
```

---

standard_exponential/1

Returns a standard exponential distributed random value.

Compilation flags:
```
static
```

Template:
```
standard_exponential(Value)
```
Mode and number of proofs:
```
standard_exponential(-float) - one
```

---

standard_gamma/2

Returns a standard gamma distributed random value.

Compilation flags:
`static`

Template:
`standard_gamma(Shape,Value)`
Mode and number of proofs:
`standard_gamma(+positive_float,-float)` - `one`

---

standard_normal/1

Returns a standard normally (Gaussian) distributed random value (using a default mean of 0.0 and a default deviation of 1.0).

Compilation flags:
`static`

Template:
`standard_normal(Value)`
Mode and number of proofs:
`standard_normal(-float)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

random_protocol, *pseudo_random_protocol*

1.269 random_forest_classifier

object

1.269.1 `random_forest_classifier`

Random Forest classifier using C4.5 decision trees as base learners. Builds an ensemble of decision trees trained on bootstrap samples with random feature subsets and combines their predictions through majority voting.

Availability:
`logtalk_load(random_forest_classifier(loader))`

Author: Paulo Moura
Version: 2:1:1
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *probabilistic_classifier_common*

Uses:
c45_classifier
fast_random(Algorithm)
format
list
pairs
type

Remarks:
(none)

Inherited public predicates:
check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *c45_classifier*, *isolation_forest_anomaly_detector*, *knn_classifier*, *naive_bayes_classifier*, *nearest_centroid_classifier*, *adaptive_boosting_classifier*

1.270 random_forest_regression

object

1.270.1 `random_forest_regression`

Random Forest regression using regression trees as base learners trained on bootstrap samples and per-split random feature subsets.

Availability:

```
logtalk_load(random_forest_regression(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *regressor_common*

Uses:

fast_random(Algorithm)
format
list
population
regression_tree
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

linear_regression, *knn_regression*, *regression_tree*, *gradient_boosting_regression*

1.271 random_projection

object

1.271.1 `random_projection`

Random projection reducer for continuous datasets using a portable seeded Rademacher projection matrix.

Availability:

```
logtalk_load(random_projection(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *dimension_reducer_common*

Uses:

fast_random(Algorithm)
format
list
pairs
type

Remarks:

(none)

Inherited public predicates:

check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

lda_projection, *pca_projection*

1.272 rank_centrality

object

1.272.1 `rank_centrality`

Rank Centrality pairwise preference ranker. Learns one stationary probability score per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol by applying power iteration to the Rank Centrality Markov chain built from aggregated head-to-head outcomes, and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:

`logtalk_load(rank_centrality(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-06-06

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *ranking_dataset_common*
`public` *score_ranker_common*

Uses:

avltree
format
list

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *bradley_terry_ranker*, *copeland_ranker*

1.273 ranked_pairs

object

1.273.1 `ranked_pairs`

Ranked Pairs pairwise preference ranker. Learns one deterministic score per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol by locking direct pairwise victories in descending strength order while avoiding cycles and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:
`logtalk_load(ranked_pairs(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-06-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ranking_dataset_common*
`public` *score_ranker_model_common*
`public` *condorcet_victory_common*

Uses:

avltree
list
numberlist

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
  - *locked_pairs/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

locked_pairs/2

Returns the accepted lock graph in lock order as `lock(Item1,Item2,Strength)` terms for the learned items.

Compilation flags:

`static`

Template:

`locked_pairs(Ranker,LockedPairs)`

Mode and number of proofs:

`locked_pairs(+compound,-list(compound))` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *schulze_ranker*, *copeland_ranker*

1.274 ranking_protocols

category

1.274.1 `condorcet_victory_common`

Shared direct-victory preprocessing helpers for Condorcet-family rankers.

Availability:

```
logtalk_load(ranking_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-26

Compilation flags:

```
static
```

Uses:

avltree
list

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
  - *index_items/3*
  - *build_direct_strengths/5*

    - *zero_matrix/2*
    - *matrix_entry/4*
    - *set_matrix_entry/5*
  - *Private predicates*
  - *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

index_items/3

Builds `Item-Index` pairs for the ordered training items.

Compilation flags:
`static`

Template:
`index_items(Items,Index,Indices)`
Mode and number of proofs:
`index_items(+list,+integer,-list) - one`

---

build_direct_strengths/5

Builds the dense direct-victory strength matrix from aggregated pairwise matchups using the selected victory-strength semantics.

Compilation flags:
`static`

Template:
`build_direct_strengths(Matchups,IndexDictionary,Count,VictoryStrength,DirectStrengths)`
Mode and number of proofs:
`build_direct_strengths(+list,+compound,+integer,+atom,-list) - one`

---

zero_matrix/2

Builds a square zero matrix with the given row and column count.

Compilation flags:
`static`

Template:
`zero_matrix(Count,Matrix)`
Mode and number of proofs:
`zero_matrix(+integer,-list)` - `one`

---

matrix_entry/4

Returns the value stored at the given matrix row and column.

Compilation flags:
`static`

Template:
`matrix_entry(Matrix,RowIndex,ColumnIndex,Value)`
Mode and number of proofs:
`matrix_entry(+list,+integer,+integer,-integer)` - `one`

---

set_matrix_entry/5

Returns an updated matrix where the given row and column entry was replaced by the supplied value.

Compilation flags:
`static`

Template:
`set_matrix_entry(Matrix,RowIndex,ColumnIndex,Value,UpdatedMatrix)`
Mode and number of proofs:
`set_matrix_entry(+list,+integer,+integer,+integer,-list)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.274.2 `glicko2_common`

Shared internal Glicko-2 numeric helpers for rating updates, volatility search, and scale conversions.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-02

Compilation flags:
`static`

Uses:
avltree

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *update_item_parameters/10*
  - *to_mu/2*
  - *to_phi/2*
  - *to_rating/2*
  - *to_deviation/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

update_item_parameters/10

Computes the simultaneous Glicko-2 rating, deviation, and volatility update for one item over a rating period using the current dictionaries and aggregated game results.

Compilation flags:

```
static
```

Template:

```
update_item_parameters(Item,ItemResults,Ratings,Deviations,Volatilities,Tau,
VolatilityTolerance,UpdatedRating,UpdatedDeviation,UpdatedVolatility)
```

Mode and number of proofs:

```
update_item_parameters(+term,+list(compound),+compound,+compound,+compound,
+number,+number,-number,-number,-number) - one
```

---

to_mu/2

Converts a rating on the external Glicko scale to the internal `mu` scale.

Compilation flags:

```
static
```

Template:

```
to_mu(Rating,Mu)
```

Mode and number of proofs:

```
to_mu(+number,-number) - one
```

---

to_phi/2

Converts a rating deviation on the external Glicko scale to the internal `phi` scale.

Compilation flags:

```
static
```

Template:

```
to_phi(Deviation,Phi)
```

Mode and number of proofs:

```
to_phi(+number,-number) - one
```

---

to_rating/2

Converts a value on the internal mu scale back to the external Glicko rating scale.

Compilation flags:
static

Template:
to_rating(Mu,Rating)
Mode and number of proofs:
to_rating(+number,-number) - one

---

to_deviation/2

Converts a value on the internal phi scale back to the external Glicko rating-deviation scale.

Compilation flags:
static

Template:
to_deviation(Phi,Deviation)
Mode and number of proofs:
to_deviation(+number,-number) - one

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.274.3 grouped_strength_ranker_common

Shared connectivity, positive-strength, and iterative-update helpers for grouped ranking models that learn one strength parameter per item.

Availability:
logtalk_load(ranking_protocols(loader))

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-25

Compilation flags:

`static`

Uses:

avltree
list

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
  - *update_strengths/4*
  - *initial_strengths/2*
  - *strength_pairs/3*
  - *optimize_strengths/8*
  - *strength_dictionary/2*
  - *require_strong_connectivity/4*
  - *strongly_connected/3*
  - *strongly_connected_components/4*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

update_strengths/4

Hook predicate that importing rankers must define to perform one synchronous strength-update iteration.

Compilation flags:

`static`

Template:

`update_strengths(Context,Strengths0,Strengths,MaximumDifference)`

Mode and number of proofs:

`update_strengths(+nonvar,+list(number),-list(number),-number)` - `one`

initial_strengths/2

Initializes a positive strength vector with a uniform value for each ordered item.

Compilation flags:

```
static
```

Template:

```
initial_strengths(Items,Strengths)
```

Mode and number of proofs:

```
initial_strengths(+list,-list(float)) - one
```

---

strength_pairs/3

Pairs ordered items with their learned strength values.

Compilation flags:

```
static
```

Template:

```
strength_pairs(Items,StrengthValues,Strengths)
```

Mode and number of proofs:

```
strength_pairs(+list,+list(number),-list(pair)) - one
```

---

optimize_strengths/8

Runs the iterative strength-update loop until convergence or the maximum iteration bound is reached.

Compilation flags:

```
static
```

Template:

```
optimize_strengths(MaximumIterations,Tolerance,UpdateContext,Strengths0,
Strengths,Status,Iterations,FinalDifference)
```

Mode and number of proofs:

```
optimize_strengths(+integer,+number,+nonvar,+list(number),-list(number),-atom,
-integer,-number) - one
```

---

strength_dictionary/2

Builds an index-addressable dictionary from an ordered list of strength values.

Compilation flags:
```
static
```

Template:
```
strength_dictionary(Strengths,Dictionary)
```
Mode and number of proofs:
```
strength_dictionary(+list(number),-compound) - one
```

---

require_strong_connectivity/4

Checks that the directed graph is strongly connected and throws a domain error with the strongly connected components otherwise.

Compilation flags:
```
static
```

Template:
```
require_strong_connectivity(Count,DirectedAdjacency,ReverseAdjacency,ErrorDomain)
```
Mode and number of proofs:
```
require_strong_connectivity(+integer,+compound,+compound,+atom) - one
```

---

strongly_connected/3

True when all indexed items are mutually reachable in both the forward and reverse adjacency graphs.

Compilation flags:
```
static
```

Template:
```
strongly_connected(Count,DirectedAdjacency,ReverseAdjacency)
```
Mode and number of proofs:
```
strongly_connected(+integer,+compound,+compound) - zero_or_one
```

`strongly_connected_components/4`

Computes the strongly connected components of the directed graph induced by the forward and reverse adjacency dictionaries.

Compilation flags:
`static`

Template:
`strongly_connected_components(Count,DirectedAdjacency,ReverseAdjacency,Components)`

Mode and number of proofs:
`strongly_connected_components(+integer,+compound,+compound,-list)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.274.4 `multiplayer_ranking_dataset_protocol`

Protocol for ordered multiplayer ranking datasets with ranked teams and weighted player participation.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-02

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

    - *item/1*
    - *match/1*
    - *team/3*
    - *team_member/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

item/1

Enumerates by backtracking the items that can be ranked.

Compilation flags:
```
static
```

Template:
```
item(Item)
```
Mode and number of proofs:
```
item(-atom) - zero_or_more
```

---

match/1

Enumerates by backtracking the matches in chronological processing order.

Compilation flags:
```
static
```

Template:
```
match(Match)
```
Mode and number of proofs:
```
match(-atom) - zero_or_more
```

---

team/3

Enumerates by backtracking the teams in a match and their non-negative ranks. Lower ranks are better and equal ranks denote a draw.

Compilation flags:
```
static
```

Template:
```
team(Match,Team,Rank)
```

Mode and number of proofs:
`team(-atom,-atom,-integer) - zero_or_more`

`team_member/4`

Enumerates by backtracking team members and their participation weights, which must be greater than zero and no greater than one.

Compilation flags:
`static`

Template:
`team_member(Match,Team,Item,Weight)`
Mode and number of proofs:
`team_member(-atom,-atom,-atom,-number) - zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.274.5 `pairwise_measurement_dataset_protocol`

Protocol for pairwise measurement datasets.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *item/1*
  - *measurement/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

item/1

Enumerates by backtracking the items that can be ranked.

Compilation flags:
`static`

Template:
`item(Item)`
Mode and number of proofs:
`item(-atom)` - `zero_or_more`

---

measurement/4

Enumerates by backtracking weighted pairwise signed measurements where `Value` is the observed scalar measurement for the oriented edge `Item1 -> Item2` and `Weight` is its positive confidence or repetition weight.

Compilation flags:
`static`

Template:
`measurement(Item1,Item2,Value,Weight)`
Mode and number of proofs:
`measurement(-atom,-atom,-number,-number)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.274.6 `pairwise_ranking_dataset_protocol`

Protocol for pairwise ranking datasets.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *item/1*
  - *preference/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

item/1

Enumerates by backtracking the items that can be ranked.

Compilation flags:
`static`

Template:
`item(Item)`
Mode and number of proofs:
`item(-atom)` - `zero_or_more`

---

preference/3

Enumerates by backtracking weighted pairwise preferences where `Winner` is preferred over `Loser` `Weight` times.

Compilation flags:
`static`

Template:
`preference(Winner,Loser,Weight)`
Mode and number of proofs:
`preference(-atom,-atom,-number)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.274.7 pairwise_strength_ranker_common

Shared preprocessing, connectivity, and MM-iteration helpers for pairwise strength rankers.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-04-25

Compilation flags:
`static`

Uses:
avltree
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *update_strengths/4*
  - *initial_strengths/2*
  - *strength_pairs/3*
  - *index_items/3*
  - *preprocess_matchups/6*
  - *build_dataset_model/5*
  - *require_strong_connectivity/4*
  - *strongly_connected/3*
  - *strongly_connected_components/4*
  - *optimize_strengths/8*
  - *strength_dictionary/2*
  - *item_denominator/5*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

update_strengths/4

Hook predicate that importing rankers must define to perform one synchronous strength-update iteration.

Compilation flags:
`static`

Template:
`update_strengths(Context,Strengths0,Strengths,MaximumDifference)`

Mode and number of proofs:
`update_strengths(+nonvar,+list(number),-list(number),-number)` - `one`

---

initial_strengths/2

Initializes a positive strength vector with a uniform value for each ordered item.

Compilation flags:
`static`

Template:
`initial_strengths(Items,Strengths)`
Mode and number of proofs:
`initial_strengths(+list,-list(float))` - `one`

---

strength_pairs/3

Pairs ordered items with their learned strength values.

Compilation flags:
`static`

Template:
`strength_pairs(Items,StrengthValues,Strengths)`
Mode and number of proofs:
`strength_pairs(+list,+list(number),-list(pair))` - `one`

---

index_items/3

Builds `Item-Index` pairs for the ordered training items.

Compilation flags:
`static`

Template:
`index_items(Items,Index,Indices)`
Mode and number of proofs:
`index_items(+list,+integer,-list(pair))` - `one`

---

preprocess_matchups/6

Builds directed adjacency, reverse adjacency, per-item win totals, and undirected pair-adjacency structures from aggregated pairwise matchups.

Compilation flags:

```
static
```

Template:

```
preprocess_matchups(Matchups,IndexDictionary,DirectedAdjacency,ReverseAdjacency,
WinWeights,PairAdjacency)
```

Mode and number of proofs:

```
preprocess_matchups(+list,+compound,-compound,-compound,-compound,-compound) - one
```

---

build_dataset_model/5

Materializes dense per-item win totals and weighted undirected adjacency lists from the sparse preprocessing dictionaries.

Compilation flags:

```
static
```

Template:

```
build_dataset_model(Count,WinWeights,PairAdjacency,PairWeights,Wins)
```

Mode and number of proofs:

```
build_dataset_model(+integer,+compound,+compound,-list,-list(number)) - one
```

---

require_strong_connectivity/4

Checks that the directed graph is strongly connected and throws a domain error with the strongly connected components otherwise.

Compilation flags:

```
static
```

Template:

```
require_strong_connectivity(Count,DirectedAdjacency,ReverseAdjacency,ErrorDomain)
```

Mode and number of proofs:

```
require_strong_connectivity(+integer,+compound,+compound,+atom) - one
```

---

strongly_connected/3

True when all indexed items are mutually reachable in both the forward and reverse adjacency graphs.

Compilation flags:
```
static
```

Template:
```
strongly_connected(Count,DirectedAdjacency,ReverseAdjacency)
```
Mode and number of proofs:
```
strongly_connected(+integer,+compound,+compound) - zero_or_one
```

---

strongly_connected_components/4

Computes the strongly connected components of the directed graph induced by the forward and reverse adjacency dictionaries.

Compilation flags:
```
static
```

Template:
```
strongly_connected_components(Count,DirectedAdjacency,ReverseAdjacency,
Components)
```
Mode and number of proofs:
```
strongly_connected_components(+integer,+compound,+compound,-list) - one
```

---

optimize_strengths/8

Runs the iterative strength-update loop until convergence or the maximum iteration bound is reached.

Compilation flags:
```
static
```

Template:
```
optimize_strengths(MaximumIterations,Tolerance,UpdateContext,Strengths0,
Strengths,Status,Iterations,FinalDifference)
```
Mode and number of proofs:
```
optimize_strengths(+integer,+number,+nonvar,+list(number),-list(number),-atom,
-integer,-number) - one
```

---

strength_dictionary/2

Builds an index-addressable dictionary from an ordered list of strength values.

Compilation flags:
```
static
```

Template:
```
strength_dictionary(Strengths,Dictionary)
```
Mode and number of proofs:
```
strength_dictionary(+list(number),-compound) - one
```

---

item_denominator/5

Accumulates the pairwise MM denominator for one item using the current strength dictionary and weighted neighbors.

Compilation flags:
```
static
```

Template:
```
item_denominator(Neighbors,StrengthDictionary,CurrentStrength,Denominator0,
Denominator)
```
Mode and number of proofs:
```
item_denominator(+list(pair),+compound,+number,+number,-number) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.274.8 plackett_luce_common

Shared helpers for grouped Plackett-Luce rankers.

Availability:
```
logtalk_load(ranking_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-26

Compilation flags:

`static`

Uses:
avltree
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *index_items/3*
  - *preprocess_groups/10*
  - *selection_exposures/4*
  - *normalize_strengths/6*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

index_items/3

Builds `Item-Index` pairs for the ordered training items.

Compilation flags:
`static`

Template:
`index_items(Items,Index,Indices)`
Mode and number of proofs:
`index_items(+list,+integer,-list)` - `one`

preprocess_groups/10

Builds per-group choice steps, selection counts, and forward and reverse adjacency dictionaries from grouped rankings under the selected traversal order.

Compilation flags:

```
static
```

Template:

```
preprocess_groups(Order,Groups,Dataset,MissingRelevance,IndexDictionary,Count,
Steps,SelectionCounts,DirectedAdjacency,ReverseAdjacency)
```

Mode and number of proofs:

```
preprocess_groups(+atom,+list,+object_identifier,+atom,+compound,+integer,-list,
-list,-compound,-compound) - one
```

---

selection_exposures/4

Computes the per-item exposure counts implied by the grouped choice steps and the current value dictionary.

Compilation flags:

```
static
```

Template:

```
selection_exposures(Steps,SelectionCounts,ValueDictionary,ExposureCounts)
```

Mode and number of proofs:

```
selection_exposures(+list,+list,+compound,-list) - one
```

---

normalize_strengths/6

Normalizes raw strength values to sum to one while tracking the maximum absolute change from the current strengths.

Compilation flags:

```
static
```

Template:

```
normalize_strengths(RawStrengths,TotalRawStrength,CurrentStrengths,Strengths,
MaximumDifference0,MaximumDifference)
```

Mode and number of proofs:

```
normalize_strengths(+list(number),+number,+list(number),-list(number),+number,
-number) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.274.9 `ranker_common`

Shared predicates for ranker score access, diagnostics, and export.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-30

Compilation flags:
`static`

Implements:
`public` *ranker_protocol*
Extends:
`public` *options*
Uses:
avltree
format
list
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *valid_option/1* *valid_options/1* *valid_ranker/1*

- *Public predicates*
- *Protected predicates*
  - *ranker_scores_data/2*
  - *ranker_diagnostics_data/2*
  - *ranker_export_template/4*
  - *ranker_term_template/2*
  - *print_ranker_template/1*

    - *valid_ranker_metadata/2*
    - *valid_item_value_pairs/2*
  - *Private predicates*
  - *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

ranker_scores_data/2

Hook predicate that importing ranker implementations must define in order to expose the learned item-score pairs on the ranker native numeric scale.

Compilation flags:

```
static
```

Template:

```
ranker_scores_data(Ranker,Scores)
```

Mode and number of proofs:

```
ranker_scores_data(+compound,-list(pair)) - one
```

---

ranker_diagnostics_data/2

Hook predicate that importing ranker implementations must define in order to expose diagnostics metadata.

Compilation flags:

```
static
```

Template:

```
ranker_diagnostics_data(Ranker,Diagnostics)
```

Mode and number of proofs:

```
ranker_diagnostics_data(+compound,-list(compound)) - one
```

---

ranker_export_template/4

Hook predicate that importing ranker implementations must define in order to expose the exported ranker template for a given functor.

Compilation flags:

```
static
```

Template:

```
ranker_export_template(Dataset,Ranker,Functor,Template)
```

Mode and number of proofs:

```
ranker_export_template(+object_identifier,+compound,+atom,-callable) - one
```

---

ranker_term_template/2

Hook predicate that importing ranker implementations must define in order to expose the learned ranker term template used by pretty-printing helpers.

Compilation flags:

```
static
```

Template:

```
ranker_term_template(Ranker,Template)
```

Mode and number of proofs:

```
ranker_term_template(+compound,-callable) - one
```

---

print_ranker_template/1

Pretty-printing helper predicate used by importing ranker implementations to show the learned ranker term template.

Compilation flags:

```
static
```

Template:

```
print_ranker_template(Ranker)
```

Mode and number of proofs:

```
print_ranker_template(+compound) - one
```

---

valid_ranker_metadata/2

True when the diagnostics list contains the expected model identifier, a valid effective options list, and a structurally valid dataset summary.

Compilation flags:

```
static
```

Template:

```
valid_ranker_metadata(Model,Diagnostics)
```

Mode and number of proofs:

```
valid_ranker_metadata(+atom,+list(compound)) - zero_or_one
```

---

valid_item_value_pairs/2

True when the second argument is an ordered list of unique `Item-Value` pairs aligned with the first argument item list.

Compilation flags:
`static`

Template:
`valid_item_value_pairs(Items,Pairs)`
Mode and number of proofs:
`valid_item_value_pairs(+list,+list(pair))` - `zero_or_one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.274.10 ranker_protocol

Protocol for machine learning rankers.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/3*

      - *learn/2*
      - *rank/3*
      - *scores/2*
      - *diagnostics/2*
      - *diagnostic/2*
      - *ranker_options/2*
      - *check_ranker/1*
      - *valid_ranker/1*
      - *export_to_clauses/4*
      - *export_to_file/4*
      - *print_ranker/1*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

learn/3

Learns a ranker from the given dataset object using the specified options.

Compilation flags:
`static`

Template:
`learn(Dataset,Ranker,Options)`

Mode and number of proofs:
`learn(+object_identifier,-compound,+list(compound))` - `one`

---

learn/2

Learns a ranker from the given dataset object using default options.

Compilation flags:
`static`

Template:
`learn(Dataset,Ranker)`

Mode and number of proofs:
`learn(+object_identifier,-compound)` - `one`

---

rank/3

Ranks a list of candidate items using the learned ranker and returns the candidates ordered from most preferred to least preferred.

Compilation flags:

```
static
```

Template:

```
rank(Ranker,Candidates,Ranking)
```

Mode and number of proofs:

```
rank(+compound,+list,-list) - one
```

---

scores/2

Returns the learned item-score pairs on the ranker native numeric scale. Depending on the ranker, these values may also be interpreted more specifically as strengths or ratings.

Compilation flags:

```
static
```

Template:

```
scores(Ranker,Scores)
```

Mode and number of proofs:

```
scores(+compound,-list(pair)) - one
```

---

diagnostics/2

Returns diagnostics and metadata associated with a learned ranker in a representation-independent way.

Compilation flags:

```
static
```

Template:

```
diagnostics(Ranker,Diagnostics)
```

Mode and number of proofs:

```
diagnostics(+compound,-list(compound)) - one
```

---

diagnostic/2

Tests or enumerates individual diagnostics metadata terms for a learned ranker.

Compilation flags:
`static`

Template:
`diagnostic(Ranker,Diagnostic)`
Mode and number of proofs:
`diagnostic(+compound,?compound)` - `zero_or_more`

---

ranker_options/2

Returns the effective training options recorded in a learned ranker diagnostics list.

Compilation flags:
`static`

Template:
`ranker_options(Ranker,Options)`
Mode and number of proofs:
`ranker_options(+compound,-list(compound))` - `zero_or_one`

---

check_ranker/1

Checks that the argument is a structurally valid learned ranker term for the receiving ranker implementation, throwing an exception on invalid input when applicable.

Compilation flags:
`static`

Template:
`check_ranker(Ranker)`
Mode and number of proofs:
`check_ranker(@compound)` - `one_or_error`

Exceptions:
`Ranker` is a variable:
`instantiation_error`
`Ranker` is neither a variable nor a valid ranker:
`domain_error(ranker,Ranker)`

---

valid_ranker/1

True when *check_ranker/1* succeeds for the argument without throwing an exception.

Compilation flags:
`static`

Template:
`valid_ranker(Ranker)`
Mode and number of proofs:
`valid_ranker(@compound)` - `zero_or_one`

---

export_to_clauses/4

Converts a ranker into a list of predicate clauses. `Functor` is the functor for the generated predicate clauses. When exporting a serialized ranker term, a noun such as `ranker` or `model` is usually clearer than a verb or result label.

Compilation flags:
`static`

Template:
`export_to_clauses(Dataset,Ranker,Functor,Clauses)`
Mode and number of proofs:
`export_to_clauses(+object_identifier,+compound,+callable,-list(clause))` - `one`

---

export_to_file/4

Exports a ranker to a file. `Functor` is the functor for the generated predicate clauses. When exporting a serialized ranker term, a noun such as `ranker` or `model` is usually clearer than a verb or result label.

Compilation flags:
`static`

Template:
`export_to_file(Dataset,Ranker,Functor,File)`
Mode and number of proofs:
`export_to_file(+object_identifier,+compound,+callable,+atom)` - `one`

`print_ranker/1`

Prints a ranker to the current output stream in a human-readable format.

Compilation flags:
`static`

Template:
`print_ranker(Ranker)`
Mode and number of proofs:
`print_ranker(+compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

bradley_terry_ranker, *ranking_dataset_protocol*, *pairwise_ranking_dataset_protocol*

category

1.274.11 `ranking_dataset_common`

Shared predicates for collecting, analyzing, and validating ranking datasets.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static`

Uses:
avltree
list
numberlist

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *pairwise_dataset_declared_items/2*
  - *pairwise_dataset_items/2*
  - *pairwise_dataset_preferences/2*
  - *pairwise_dataset_connected_components/2*
  - *pairwise_dataset_summary/2*
  - *pairwise_dataset_win_totals/2*
  - *pairwise_dataset_matchups/2*
  - *pairwise_measurement_dataset_declared_items/2*
  - *pairwise_measurement_dataset_items/2*
  - *pairwise_measurement_dataset_measurements/2*
  - *pairwise_measurement_dataset_connected_components/2*
  - *pairwise_measurement_dataset_summary/2*
  - *temporal_pairwise_dataset_declared_items/2*
  - *temporal_pairwise_dataset_items/2*
  - *temporal_pairwise_dataset_periods/2*
  - *temporal_pairwise_dataset_games/2*
  - *temporal_pairwise_dataset_games/3*
  - *temporal_pairwise_dataset_connected_components/2*
  - *temporal_pairwise_dataset_summary/2*
  - *multiplayer_dataset_declared_items/2*
  - *multiplayer_dataset_items/2*
  - *multiplayer_dataset_declared_matches/2*
  - *multiplayer_dataset_matches/2*
  - *multiplayer_dataset_match_teams/3*
  - *multiplayer_dataset_events/2*
  - *multiplayer_dataset_connected_components/2*
  - *multiplayer_dataset_summary/2*
  - *grouped_dataset_groups/2*
  - *grouped_dataset_items/2*
  - *grouped_dataset_items/3*
  - *grouped_dataset_summary/2*
  - *validate_pairwise_dataset/1*
  - *validate_pairwise_dataset/2*
  - *validate_pairwise_measurement_dataset/1*
  - *validate_pairwise_measurement_dataset/2*
  - *validate_grouped_dataset/1*
  - *validate_temporal_pairwise_dataset/1*
  - *validate_temporal_pairwise_dataset/2*
  - *validate_multiplayer_dataset/1*
  - *validate_multiplayer_dataset/2*

  - *validate_grouped_dataset/2*
- *Protected predicates*
  - *grouped_dataset_item_relevances/4*
  - *grouped_dataset_relevance_frequencies/2*
  - *grouped_dataset_item_relevance_scores/3*
  - *grouped_dataset_tie_blocks/4*
- *Private predicates*
- *Operators*

Public predicates

pairwise_dataset_declared_items/2

Collects the pairwise dataset declared items preserving declaration order.

Compilation flags:
`static`

Template:
`pairwise_dataset_declared_items(Dataset,Items)`
Mode and number of proofs:
`pairwise_dataset_declared_items(+object_identifier,-list)` - `one`

---

pairwise_dataset_items/2

Collects the unique pairwise dataset items preserving their first declaration order.

Compilation flags:
`static`

Template:
`pairwise_dataset_items(Dataset,Items)`
Mode and number of proofs:
`pairwise_dataset_items(+object_identifier,-list)` - `one`

---

pairwise_dataset_preferences/2

Collects the pairwise dataset preferences as `p(Winner,Loser,Weight)` terms.

Compilation flags:
`static`

Template:

```
pairwise_dataset_preferences(Dataset,Preferences)
```

Mode and number of proofs:

```
pairwise_dataset_preferences(+object_identifier,-list) - one
```

---

pairwise_dataset_connected_components/2

Returns the connected components induced by the pairwise preference graph.

Compilation flags:

```
static
```

Template:

```
pairwise_dataset_connected_components(Dataset,Components)
```

Mode and number of proofs:

```
pairwise_dataset_connected_components(+object_identifier,-list(list)) - one
```

---

pairwise_dataset_summary/2

Returns a summary of a pairwise dataset, including item, preference, component, and isolated-item counts.

Compilation flags:

```
static
```

Template:

```
pairwise_dataset_summary(Dataset,Summary)
```

Mode and number of proofs:

```
pairwise_dataset_summary(+object_identifier,-list(compound)) - one
```

---

pairwise_dataset_win_totals/2

Returns per-item total win weights for a pairwise dataset.

Compilation flags:

```
static
```

Template:

```
pairwise_dataset_win_totals(Dataset,Totals)
```

Mode and number of proofs:

```
pairwise_dataset_win_totals(+object_identifier,-list(pair)) - one
```

---

pairwise_dataset_matchups/2

Returns aggregated pairwise matchups as `matchup(Item1,Item2,Item1Wins,Item2Wins)` terms preserving item declaration order inside each pair.

Compilation flags:
`static`

Template:
`pairwise_dataset_matchups(Dataset,Matchups)`
Mode and number of proofs:
`pairwise_dataset_matchups(+object_identifier,-list(compound))` - `one`

---

pairwise_measurement_dataset_declared_items/2

Collects the pairwise measurement dataset declared items preserving declaration order.

Compilation flags:
`static`

Template:
`pairwise_measurement_dataset_declared_items(Dataset,Items)`
Mode and number of proofs:
`pairwise_measurement_dataset_declared_items(+object_identifier,-list)` - `one`

---

pairwise_measurement_dataset_items/2

Collects the unique pairwise measurement dataset items preserving their first declaration order.

Compilation flags:
`static`

Template:
`pairwise_measurement_dataset_items(Dataset,Items)`
Mode and number of proofs:
`pairwise_measurement_dataset_items(+object_identifier,-list)` - `one`

---

pairwise_measurement_dataset_measurements/2

Collects the pairwise measurement dataset observations as `m(Item1,Item2,Value,Weight)` terms preserving enumeration order.

Compilation flags:

```
static
```

Template:

```
pairwise_measurement_dataset_measurements(Dataset,Measurements)
```

Mode and number of proofs:

```
pairwise_measurement_dataset_measurements(+object_identifier,-list(compound)) -
one
```

---

pairwise_measurement_dataset_connected_components/2

Returns the connected components induced by the pairwise measurement support graph.

Compilation flags:

```
static
```

Template:

```
pairwise_measurement_dataset_connected_components(Dataset,Components)
```

Mode and number of proofs:

```
pairwise_measurement_dataset_connected_components(+object_identifier,-list(list))
- one
```

---

pairwise_measurement_dataset_summary/2

Returns a summary of a pairwise measurement dataset, including item, measurement, component, and isolated-item counts.

Compilation flags:

```
static
```

Template:

```
pairwise_measurement_dataset_summary(Dataset,Summary)
```

Mode and number of proofs:

```
pairwise_measurement_dataset_summary(+object_identifier,-list(compound)) - one
```

---

temporal_pairwise_dataset_declared_items/2

Collects the temporal pairwise dataset declared items preserving declaration order.

Compilation flags:
```
static
```

Template:
```
temporal_pairwise_dataset_declared_items(Dataset,Items)
```
Mode and number of proofs:
```
temporal_pairwise_dataset_declared_items(+object_identifier,-list) - one
```

---

temporal_pairwise_dataset_items/2

Collects the unique temporal pairwise dataset items preserving their first declaration order.

Compilation flags:
```
static
```

Template:
```
temporal_pairwise_dataset_items(Dataset,Items)
```
Mode and number of proofs:
```
temporal_pairwise_dataset_items(+object_identifier,-list) - one
```

---

temporal_pairwise_dataset_periods/2

Collects the unique temporal pairwise dataset periods preserving their first declaration order.

Compilation flags:
```
static
```

Template:
```
temporal_pairwise_dataset_periods(Dataset,Periods)
```
Mode and number of proofs:
```
temporal_pairwise_dataset_periods(+object_identifier,-list) - one
```

---

temporal_pairwise_dataset_games/2

Collects the temporal pairwise dataset games as `game(Period,Item1,Item2,Score)` terms preserving enumeration order.

Compilation flags:
`static`

Template:
`temporal_pairwise_dataset_games(Dataset,Games)`
Mode and number of proofs:
`temporal_pairwise_dataset_games(+object_identifier,-list(compound))` - `one`

---

temporal_pairwise_dataset_games/3

Collects the temporal pairwise dataset games for a single period as `game(Item1,Item2,Score)` terms preserving enumeration order.

Compilation flags:
`static`

Template:
`temporal_pairwise_dataset_games(Dataset,Period,Games)`
Mode and number of proofs:
`temporal_pairwise_dataset_games(+object_identifier,+term,-list(compound))` - `one`

---

temporal_pairwise_dataset_connected_components/2

Returns the connected components induced by the temporal pairwise game graph.

Compilation flags:
`static`

Template:
`temporal_pairwise_dataset_connected_components(Dataset,Components)`
Mode and number of proofs:
`temporal_pairwise_dataset_connected_components(+object_identifier,-list(list))` - `one`

---

temporal_pairwise_dataset_summary/2

Returns a summary of a temporal pairwise dataset, including item, period, game, component, and isolated-item counts.

Compilation flags:

```
static
```

Template:

```
temporal_pairwise_dataset_summary(Dataset,Summary)
```

Mode and number of proofs:

```
temporal_pairwise_dataset_summary(+object_identifier,-list(compound)) - one
```

---

multiplayer_dataset_declared_items/2

Collects the multiplayer dataset declared items preserving declaration order.

Compilation flags:

```
static
```

Template:

```
multiplayer_dataset_declared_items(Dataset,Items)
```

Mode and number of proofs:

```
multiplayer_dataset_declared_items(+object_identifier,-list) - one
```

---

multiplayer_dataset_items/2

Collects the unique multiplayer dataset items preserving their first declaration order.

Compilation flags:

```
static
```

Template:

```
multiplayer_dataset_items(Dataset,Items)
```

Mode and number of proofs:

```
multiplayer_dataset_items(+object_identifier,-list) - one
```

---

multiplayer_dataset_declared_matches/2

Collects the multiplayer dataset declared matches preserving chronological declaration order.

Compilation flags:
`static`

Template:
`multiplayer_dataset_declared_matches(Dataset,Matches)`
Mode and number of proofs:
`multiplayer_dataset_declared_matches(+object_identifier,-list)` - `one`

---

multiplayer_dataset_matches/2

Collects the unique multiplayer dataset matches preserving their first declaration order.

Compilation flags:
`static`

Template:
`multiplayer_dataset_matches(Dataset,Matches)`
Mode and number of proofs:
`multiplayer_dataset_matches(+object_identifier,-list)` - `one`

---

multiplayer_dataset_match_teams/3

Materializes one match as ordered `team(Team,Rank,Members)` terms, where members are `Item-Weight` pairs.

Compilation flags:
`static`

Template:
`multiplayer_dataset_match_teams(Dataset,Match,Teams)`
Mode and number of proofs:
`multiplayer_dataset_match_teams(+object_identifier,+term,-list(compound))` - `one`

---

multiplayer_dataset_events/2

Materializes matches in chronological order as `match(Match,Teams)` terms.

Compilation flags:
`static`

Template:
`multiplayer_dataset_events(Dataset,Events)`
Mode and number of proofs:
`multiplayer_dataset_events(+object_identifier,-list(compound))` - `one`

---

multiplayer_dataset_connected_components/2

Returns the connected components induced by players participating in the same match.

Compilation flags:
`static`

Template:
`multiplayer_dataset_connected_components(Dataset,Components)`
Mode and number of proofs:
`multiplayer_dataset_connected_components(+object_identifier,-list(list))` - `one`

---

multiplayer_dataset_summary/2

Returns a summary of a multiplayer dataset, including item, match, team, participation, component, and isolated-item counts.

Compilation flags:
`static`

Template:
`multiplayer_dataset_summary(Dataset,Summary)`
Mode and number of proofs:
`multiplayer_dataset_summary(+object_identifier,-list(compound))` - `one`

---

grouped_dataset_groups/2

Collects the unique grouped-dataset groups preserving their first declaration order.

Compilation flags:
```
static
```

Template:
```
grouped_dataset_groups(Dataset,Groups)
```
Mode and number of proofs:
```
grouped_dataset_groups(+object_identifier,-list) - one
```

---

grouped_dataset_items/2

Collects the unique items declared across all groups in a grouped dataset preserving their first declaration order.

Compilation flags:
```
static
```

Template:
```
grouped_dataset_items(Dataset,Items)
```
Mode and number of proofs:
```
grouped_dataset_items(+object_identifier,-list) - one
```

---

grouped_dataset_items/3

Collects the unique items declared for a grouped-dataset group.

Compilation flags:
```
static
```

Template:
```
grouped_dataset_items(Dataset,Group,Items)
```
Mode and number of proofs:
```
grouped_dataset_items(+object_identifier,+term,-list) - one
```

---

grouped_dataset_summary/2

Returns a summary of a grouped dataset, including group, item, and relevance-judgment counts.

Compilation flags:

```
static
```

Template:

```
grouped_dataset_summary(Dataset,Summary)
```

Mode and number of proofs:

```
grouped_dataset_summary(+object_identifier,-list(compound)) - one
```

---

validate_pairwise_dataset/1

Validates a pairwise ranking dataset and throws an error if the dataset is malformed or disconnected.

Compilation flags:

```
static
```

Template:

```
validate_pairwise_dataset(Dataset)
```

Mode and number of proofs:

```
validate_pairwise_dataset(+object_identifier) - one
```

---

validate_pairwise_dataset/2

Validates a pairwise ranking dataset and returns its dataset summary when validation succeeds.

Compilation flags:

```
static
```

Template:

```
validate_pairwise_dataset(Dataset,Summary)
```

Mode and number of proofs:

```
validate_pairwise_dataset(+object_identifier,-list(compound)) - one
```

---

validate_pairwise_measurement_dataset/1

Validates a pairwise measurement dataset and throws an error if the dataset is malformed or disconnected.

Compilation flags:

```
static
```

Template:

```
validate_pairwise_measurement_dataset(Dataset)
```

Mode and number of proofs:

```
validate_pairwise_measurement_dataset(+object_identifier) - one
```

---

validate_pairwise_measurement_dataset/2

Validates a pairwise measurement dataset and returns its dataset summary when validation succeeds.

Compilation flags:

```
static
```

Template:

```
validate_pairwise_measurement_dataset(Dataset,Summary)
```

Mode and number of proofs:

```
validate_pairwise_measurement_dataset(+object_identifier,-list(compound)) - one
```

---

validate_grouped_dataset/1

Validates a grouped ranking dataset and throws an error if the dataset is malformed.

Compilation flags:

```
static
```

Template:

```
validate_grouped_dataset(Dataset)
```

Mode and number of proofs:

```
validate_grouped_dataset(+object_identifier) - one
```

---

validate_temporal_pairwise_dataset/1

Validates a temporal pairwise ranking dataset and throws an error if the dataset is malformed or disconnected.

Compilation flags:
```
static
```

Template:
```
validate_temporal_pairwise_dataset(Dataset)
```
Mode and number of proofs:
```
validate_temporal_pairwise_dataset(+object_identifier) - one
```

---

validate_temporal_pairwise_dataset/2

Validates a temporal pairwise ranking dataset and returns its dataset summary when validation succeeds.

Compilation flags:
```
static
```

Template:
```
validate_temporal_pairwise_dataset(Dataset,Summary)
```
Mode and number of proofs:
```
validate_temporal_pairwise_dataset(+object_identifier,-list(compound)) - one
```

---

validate_multiplayer_dataset/1

Validates a multiplayer ranking dataset and throws an error if the dataset is malformed.

Compilation flags:
```
static
```

Template:
```
validate_multiplayer_dataset(Dataset)
```
Mode and number of proofs:
```
validate_multiplayer_dataset(+object_identifier) - one
```

---

validate_multiplayer_dataset/2

Validates a multiplayer ranking dataset and returns its dataset summary when validation succeeds.

Compilation flags:

```
static
```

Template:

```
validate_multiplayer_dataset(Dataset,Summary)
```

Mode and number of proofs:

```
validate_multiplayer_dataset(+object_identifier,-list(compound)) - one
```

---

validate_grouped_dataset/2

Validates a grouped ranking dataset and returns its dataset summary when validation succeeds.

Compilation flags:

```
static
```

Template:

```
validate_grouped_dataset(Dataset,Summary)
```

Mode and number of proofs:

```
validate_grouped_dataset(+object_identifier,-list(compound)) - one
```

Protected predicates

grouped_dataset_item_relevances/4

Collects the declared items for a grouped-dataset group paired with their relevance values under the given missing-relevance policy.

Compilation flags:

```
static
```

Template:

```
grouped_dataset_item_relevances(Dataset,Group,MissingRelevance,ItemRelevances)
```

Mode and number of proofs:

```
grouped_dataset_item_relevances(+object_identifier,+term,+atom,-list(pair)) - one
```

---

grouped_dataset_relevance_frequencies/2

Aggregates grouped item relevances into a relevance-frequency dictionary.

Compilation flags:
`static`

Template:
`grouped_dataset_relevance_frequencies(ItemRelevances,FrequencyDictionary)`
Mode and number of proofs:
`grouped_dataset_relevance_frequencies(+list(pair),-compound)` - `one`

---

grouped_dataset_item_relevance_scores/3

Maps grouped item relevances to per-item score deltas using a relevance-points dictionary.

Compilation flags:
`static`

Template:
`grouped_dataset_item_relevance_scores(ItemRelevances,PointsDictionary,ItemScores)`
Mode and number of proofs:
`grouped_dataset_item_relevance_scores(+list(pair),+compound,-list(pair))` - `one`

---

grouped_dataset_tie_blocks/4

Collects grouped-dataset item relevances into descending relevance tie blocks represented as `tie_block(Relevance,Items)` terms, preserving the declaration order of items inside each block.

Compilation flags:
`static`

Template:
`grouped_dataset_tie_blocks(Dataset,Group,MissingRelevance,TieBlocks)`
Mode and number of proofs:
`grouped_dataset_tie_blocks(+object_identifier,+term,+atom,-list(compound))` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.274.12 `ranking_dataset_protocol`

Protocol for grouped ranking datasets.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-16

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *group/1*
  - *item/2*
  - *relevance/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

group/1

Enumerates by backtracking the ranking groups or contexts in the dataset.

Compilation flags:
```
static
```

Template:
```
group(Group)
```
Mode and number of proofs:
```
group(-atom) - zero_or_more
```

---

item/2

Enumerates by backtracking the items available inside each ranking group or context.

Compilation flags:
```
static
```

Template:
```
item(Group,Item)
```
Mode and number of proofs:
```
item(+atom,-atom) - zero_or_more
item(-atom,-atom) - zero_or_more
```

---

relevance/3

Enumerates by backtracking the non-negative integer relevance judgments associated with each group and item.

Compilation flags:
```
static
```

Template:
```
relevance(Group,Item,Relevance)
```
Mode and number of proofs:
```
relevance(+atom,+atom,-integer) - zero_or_more
relevance(-atom,-atom,-integer) - zero_or_more
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.274.13 `score_ranker_common`

Shared predicates for score-based rankers that order candidates using precomputed item scores.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-02

Compilation flags:
`static`

Extends:
`public` *ranker_common*
Uses:
avltree
list
pairs

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *valid_option/1* *valid_options/1* *valid_ranker/1*

- *Public predicates*
- *Protected predicates*
  - *rank_by_scores/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

`rank_by_scores/3`

Ranks candidates using precomputed `Item-Score` pairs while validating item existence and duplicate candidates.

Compilation flags:
`static`

Template:
`rank_by_scores(Scores,Candidates,Ranking)`
Mode and number of proofs:
`rank_by_scores(+list(pair),+list,-list)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.274.14 `score_ranker_model_common`

Shared predicates for score-ranker term construction, validation, ranking, export, and pretty-printing.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-27

Compilation flags:
`static`

Extends:
`public` *score_ranker_common*
Uses:
avltree
format
list

Remarks:

(none)

Inherited public predicates:

check_option/1 check_options/1 check_ranker/1 default_option/1 default_options/1 diagnostic/2 diagnostics/2 export_to_clauses/4 export_to_file/4 learn/2 learn/3 option/2 option/3 print_ranker/1 rank/3 ranker_options/2 scores/2 valid_option/1 valid_options/1 valid_ranker/1

- *Public predicates*
- *Protected predicates*
  - *score_ranker_model/1*
  - *score_ranker_label/1*
  - *score_ranker_term/4*
  - *valid_score/1*
  - *build_score_ranker/5*
  - *score_ranker_data/4*
  - *valid_score_ranker_diagnostics/3*
  - *initialize_scores/2*
  - *accumulate_item_scores/3*
  - *update_score/4*
  - *ordered_scores/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

score_ranker_model/1

Hook predicate that importing rankers must define to identify the learned score-ranker model.

Compilation flags:

```
static
```

Template:

```
score_ranker_model(Model)
```

Mode and number of proofs:

```
score_ranker_model(-atom) - one
```

score_ranker_label/1

Hook predicate that importing rankers must define to provide a human-readable label used by pretty-printing helpers.

Compilation flags:
`static`

Template:
`score_ranker_label(Label)`
Mode and number of proofs:
`score_ranker_label(-atom)` - `one`

---

score_ranker_term/4

Hook predicate that importing rankers must define to map between the ranker term representation and its items, scores, and diagnostics payloads.

Compilation flags:
`static`

Template:
`score_ranker_term(Items,Scores,Diagnostics,Ranker)`
Mode and number of proofs:
`score_ranker_term(?list,?list,?list(compound),?compound)` - `zero_or_one`

---

valid_score/1

Hook predicate that importing rankers must define to validate individual learned score values.

Compilation flags:
`static`

Template:
`valid_score(Score)`
Mode and number of proofs:
`valid_score(+number)` - `zero_or_one`

---

build_score_ranker/5

Builds a self-describing score-ranker term from ordered items, score pairs, effective options, and dataset summary metadata.

Compilation flags:
```
static
```

Template:
```
build_score_ranker(Items,Scores,Options,DatasetSummary,Ranker)
```
Mode and number of proofs:
```
build_score_ranker(+list,+list(pair),+list(compound),+list(compound),-compound) -
one
```

---

score_ranker_data/4

Validates and destructures a score-ranker term into its items, score pairs, and diagnostics payloads.

Compilation flags:
```
static
```

Template:
```
score_ranker_data(Ranker,Items,Scores,Diagnostics)
```
Mode and number of proofs:
```
score_ranker_data(+compound,-list,-list(pair),-list(compound)) - one
```

---

valid_score_ranker_diagnostics/3

Hook predicate that importing rankers can define to validate diagnostics beyond the common model, options, and dataset summary metadata.

Compilation flags:
```
static
```

Template:
```
valid_score_ranker_diagnostics(Items,Scores,Diagnostics)
```
Mode and number of proofs:
```
valid_score_ranker_diagnostics(+list,+list(pair),+list(compound)) - zero_or_one
```

---

initialize_scores/2

Initializes a score dictionary with zero scores for the given ordered items.

Compilation flags:
```
static
```

Template:
```
initialize_scores(Items,Scores)
```
Mode and number of proofs:
```
initialize_scores(+list,-compound) - one
```

---

accumulate_item_scores/3

Accumulates item score deltas into a score dictionary.

Compilation flags:
```
static
```

Template:
```
accumulate_item_scores(ItemDeltas,Scores,UpdatedScores)
```
Mode and number of proofs:
```
accumulate_item_scores(+list(pair),+compound,-compound) - one
```

---

update_score/4

Updates the score of a single item in a score dictionary by adding the given delta.

Compilation flags:
```
static
```

Template:
```
update_score(Scores,Item,Delta,UpdatedScores)
```
Mode and number of proofs:
```
update_score(+compound,+term,+number,-compound) - one
```

---

ordered_scores/3

Materializes ordered `Item-Score` pairs from a score dictionary using the given item order.

Compilation flags:
`static`

Template:
`ordered_scores(Items,ScoresDictionary,Scores)`
Mode and number of proofs:
`ordered_scores(+list,+compound,-list(pair))` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.274.15 temporal_pairwise_ranking_dataset_protocol

Protocol for temporal pairwise ranking datasets with explicit rating periods and per-game outcomes.

Availability:
`logtalk_load(ranking_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *item/1*

    - *period/1*
    - *game/4*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

item/1

Enumerates by backtracking the items that can be ranked.

Compilation flags:
`static`

Template:
`item(Item)`
Mode and number of proofs:
`item(-atom)` - `zero_or_more`

---

period/1

Enumerates by backtracking the rating periods in declaration order.

Compilation flags:
`static`

Template:
`period(Period)`
Mode and number of proofs:
`period(-atom)` - `zero_or_more`

---

game/4

Enumerates by backtracking observed games where `Score` is the result for `Item1` against `Item2` in the given `Period`.

Compilation flags:
`static`

Template:
`game(Period,Item1,Item2,Score)`
Mode and number of proofs:
`game(-atom,-atom,-atom,-number)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.275 reader

object

1.275.1 `reader`

Predicates for reading text file and text stream contents to lists of terms, characters, or character codes and for reading binary file and binary stream contents to lists of bytes.

Availability:
`logtalk_load(reader(loader))`

Author: Paulo Moura
Version: 2:3:0
Date: 2026-05-19

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *file_to_codes/2*
  - *file_to_codes/3*
  - *file_to_chars/2*
  - *file_to_chars/3*
  - *file_to_terms/2*
  - *file_to_terms/3*
  - *file_to_bytes/2*

      - *file_to_bytes/3*
      - *stream_to_codes/2*
      - *stream_to_codes/3*
      - *stream_to_chars/2*
      - *stream_to_chars/3*
      - *stream_to_terms/2*
      - *stream_to_terms/3*
      - *stream_to_bytes/2*
      - *stream_to_bytes/3*
      - *line_to_chars/2*
      - *line_to_chars/3*
      - *line_to_codes/2*
      - *line_to_codes/3*
      - *line_to_bytes/2*
      - *line_to_bytes/3*
    - *Protected predicates*
    - *Private predicates*
    - *Operators*

Public predicates

file_to_codes/2

Reads a text file into a list of character codes.

Compilation flags:
`static`

Template:
`file_to_codes(File,Codes)`
Mode and number of proofs:
`file_to_codes(+atom,-list(character_code))` - `one`

---

file_to_codes/3

Reads a text file into a list of character codes. The list is terminated by the given tail.

Compilation flags:
`static`

Template:
`file_to_codes(File,Codes,Tail)`
Mode and number of proofs:
`file_to_codes(+atom,-list(character_code),@term)` - `one`

---

file_to_chars/2

Reads a text file into a list of characters.

Compilation flags:
```
static
```

Template:
```
file_to_chars(File,Chars)
```
Mode and number of proofs:
```
file_to_chars(+atom,-list(character)) - one
```

---

file_to_chars/3

Reads a text file into a list of characters. The list is terminated by the given tail.

Compilation flags:
```
static
```

Template:
```
file_to_chars(File,Chars,Tail)
```
Mode and number of proofs:
```
file_to_chars(+atom,-list(character),@term) - one
```

---

file_to_terms/2

Reads a text file into a list of terms.

Compilation flags:
```
static
```

Template:
```
file_to_terms(File,Terms)
```
Mode and number of proofs:
```
file_to_terms(+atom,-list(term)) - one
```

---

file_to_terms/3

Reads a text file into a list of terms. The list is terminated by the given tail.

Compilation flags:
```
static
```

Template:
```
file_to_terms(File,Terms,Tail)
```
Mode and number of proofs:
```
file_to_terms(+atom,-list(term),@term) - one
```

---

file_to_bytes/2

Reads a binary file into a list of bytes.

Compilation flags:
```
static
```

Template:
```
file_to_bytes(File,Bytes)
```
Mode and number of proofs:
```
file_to_bytes(+atom,-list(byte)) - one
```

---

file_to_bytes/3

Reads a binary file into a list of bytes. The list is terminated by the given tail.

Compilation flags:
```
static
```

Template:
```
file_to_bytes(File,Bytes,Tail)
```
Mode and number of proofs:
```
file_to_bytes(+atom,-list(byte),@term) - one
```

---

stream_to_codes/2

Reads a text stream into a list of character codes. Does not close the stream.

Compilation flags:
```
static
```

Template:
```
stream_to_codes(Stream,Codes)
```
Mode and number of proofs:
```
stream_to_codes(+stream_or_alias,-list(character_code)) - one
```

---

stream_to_codes/3

Reads a text stream into a list of character codes. Does not close the stream. The list is terminated by the given tail.

Compilation flags:
```
static
```

Template:
```
stream_to_codes(Stream,Codes,Tail)
```
Mode and number of proofs:
```
stream_to_codes(+stream_or_alias,-list(character_code),@term) - one
```

---

stream_to_chars/2

Reads a text stream into a list of characters. Does not close the stream.

Compilation flags:
```
static
```

Template:
```
stream_to_chars(Stream,Chars)
```
Mode and number of proofs:
```
stream_to_chars(+stream_or_alias,-list(char)) - one
```

---

stream_to_chars/3

Reads a text stream into a list of characters. Does not close the stream. The list is terminated by the given tail.

Compilation flags:
```
static
```

Template:
```
stream_to_chars(Stream,Chars,Tail)
```
Mode and number of proofs:
```
stream_to_chars(+stream_or_alias,-list(char),@term) - one
```

---

stream_to_terms/2

Reads a text stream into a list of terms. Does not close the stream.

Compilation flags:
```
static
```

Template:
```
stream_to_terms(Stream,Terms)
```
Mode and number of proofs:
```
stream_to_terms(+stream_or_alias,-list(term)) - one
```

---

stream_to_terms/3

Reads a text stream into a list of terms. Does not close the stream. The list is terminated by the given tail.

Compilation flags:
```
static
```

Template:
```
stream_to_terms(Stream,Terms,Tail)
```
Mode and number of proofs:
```
stream_to_terms(+stream_or_alias,-list(term),@term) - one
```

---

stream_to_bytes/2

Reads a binary stream into a list of bytes. Does not close the stream.

Compilation flags:
```
static
```

Template:
```
stream_to_bytes(Stream,Bytes)
```
Mode and number of proofs:
```
stream_to_bytes(+stream_or_alias,-list(byte)) - one
```

---

stream_to_bytes/3

Reads a binary stream into a list of bytes. Does not close the stream. The list is terminated by the given tail.

Compilation flags:
```
static
```

Template:
```
stream_to_bytes(Stream,Bytes,Tail)
```
Mode and number of proofs:
```
stream_to_bytes(+stream_or_alias,-list(byte),@term) - one
```

---

line_to_chars/2

Reads a line from a text stream into a list of characters. Discards the end-of-line characters. Unifies `Chars` with `end_of_file` at the end of the file.

Compilation flags:
```
static
```

Template:
```
line_to_chars(Stream,Chars)
```
Mode and number of proofs:
```
line_to_chars(+stream_or_alias,-types([atom,list(character)])) - one
```

---

line_to_chars/3

Reads a line from a text stream into a list of characters. Keeps the end-of-line marker normalized to the line feed control character. The list is terminated by the given tail, which is unified with the empty list at the end of the file.

Compilation flags:

```
static
```

Template:

```
line_to_chars(Stream,Chars,Tail)
```

Mode and number of proofs:

```
line_to_chars(+stream_or_alias,-list(character),?term) - one
```

---

line_to_codes/2

Reads a line from a text stream into a list of character codes. Discards the end-of-line character codes. Unifies `Codes` with `end_of_file` at the end of the file.

Compilation flags:

```
static
```

Template:

```
line_to_codes(Stream,Codes)
```

Mode and number of proofs:

```
line_to_codes(+stream_or_alias,-types([atom,list(character_code)])) - one
```

---

line_to_codes/3

Reads a line from a text stream into a list of character codes. Keeps the end-of-line marker normalized to the line feed control character code. The list is terminated by the given tail, which is unified with the empty list at the end of the file.

Compilation flags:

```
static
```

Template:

```
line_to_codes(Stream,Codes,Tail)
```

Mode and number of proofs:

```
line_to_codes(+stream_or_alias,-list(character_code),?term) - one
```

---

line_to_bytes/2

Reads a line from a binary stream into a list of bytes. Discards the end-of-line bytes. Unifies `Bytes` with `end_of_file` at the end of the file.

Compilation flags:

```
static
```

Template:

```
line_to_bytes(Stream,Bytes)
```

Mode and number of proofs:

```
line_to_bytes(+stream_or_alias,-types([atom,list(byte)])) - one
```

---

line_to_bytes/3

Reads a line from a binary stream into a list of bytes. Keeps the end-of-line marker normalized to the line feed control byte. The list is terminated by the given tail, which is unified with the empty list at the end of the file.

Compilation flags:

```
static
```

Template:

```
line_to_bytes(Stream,Bytes,Tail)
```

Mode and number of proofs:

```
line_to_bytes(+stream_or_alias,-list(byte),?term) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.276 recommender_protocols

object

1.276.1 cosine_similarity

Cosine similarity between two sparse Key-Value vectors, using scaled unit normalization to avoid squaring original magnitudes. Norms use each whole vector and the dot product uses common keys, treating missing entries as zeros. Results range from -1.0 to 1.0, or 0.0 to 1.0 for non-negative values. No common key or an all-zero vector gives 0.0.

Availability:
`logtalk_load(recommender_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *similarity_metric_protocol*
Imports:
`public` *similarity_metric_common*

Remarks:
(none)

Inherited public predicates:
similarity/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.276.2 item_content_dataset_protocol

Protocol for item catalogs with feature occurrences or preweighted sparse content vectors.

Availability:

```
logtalk_load(recommender_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-04

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *item/1*
  - *item_content/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

item/1

Enumerates distinct instantiated atomic catalog identifiers, including items with no ratings.

Compilation flags:
`static`

Template:
`item(Item)`
Mode and number of proofs:
`item(-atomic)` - `zero_or_more`

---

item_content/2

Enumerates exactly one content descriptor for each catalog item. Descriptors are `features(Occurrences)` with ground feature occurrences or `vector(Pairs)` with unique ground feature keys and finite nonnegative weights. Empty lists are allowed. All items use the same descriptor kind.

Compilation flags:
`static`

Template:
`item_content(Item,Content)`
Mode and number of proofs:
`item_content(-atomic,-compound)` - `zero_or_more`
`item_content(+atomic,-compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

rating_dataset_protocol, *recommender_protocol*

category

1.276.3 item_content_dataset_validation

Shared collection and validation of item catalogs and content descriptors.

Availability:
`logtalk_load(recommender_protocols(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-11

Compilation flags:
`static`

Uses:
list
pairs
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *collect_contents/3*
  - *canonical_contents/3*
  - *check_rated_catalog/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

collect_contents/3

Collects and validates a nonempty catalog and its content declarations.

Compilation flags:
`static`

Template:
`collect_contents(Dataset,Contents,Kind)`

Mode and number of proofs:

`collect_contents(+object_identifier,-list(pair),-atom)` - `one_or_error`

Exceptions:

The catalog is empty:

`domain_error(non_empty_catalog,Dataset)`

An item identifier is a variable:

`instantiation_error`

An item identifier is not atomic:

`type_error(atomic,Item)`

A catalog item is declared more than once:

`domain_error(duplicate_item,Item)`

Content declarations do not cover the catalog exactly once:

`domain_error(item_content_coverage,Dataset)`

A content descriptor is invalid:

`domain_error(item_content,Content)`

Content representations are mixed:

`domain_error(content_representation,Content)`

A content list is not a proper list:

`type_error(list,List)`

A feature or content list is not a ground term:

`instantiation_error`

A sparse vector repeats a feature key:

`domain_error(duplicate_feature,Feature)`

A sparse vector entry is not a pair:

`type_error(pair,Entry)`

A sparse weight is not numeric:

`type_error(number,Weight)`

A sparse weight is negative or nonfinite:

`domain_error(non_negative_finite_weight,Weight)`

`canonical_contents/3`

Canonicalizes content declarations, preserving feature occurrences and removing zero vector weights. Fails for an empty list.

Compilation flags:

`static`

Template:

`canonical_contents(Contents,Canonical,Kind)`

Mode and number of proofs:

`canonical_contents(+list(pair),-list(pair),-atom)` - `zero_or_one_or_error`

Exceptions:

An item identifier or feature is a variable:

`instantiation_error`

An item identifier is not atomic:

```
type_error(atomic,Item)
```

An item is declared more than once:

```
domain_error(duplicate_item,Item)
```

A content descriptor is invalid:

```
domain_error(item_content,Content)
```

Content representations are mixed:

```
domain_error(content_representation,Content)
```

A content list is not a proper list:

```
type_error(list,List)
```

A sparse vector repeats a feature key:

```
domain_error(duplicate_feature,Feature)
```

A sparse vector entry is not a pair:

```
type_error(pair,Entry)
```

A sparse weight is not numeric:

```
type_error(number,Weight)
```

A sparse weight is negative or nonfinite:

```
domain_error(non_negative_finite_weight,Weight)
```

---

check_rated_catalog/2

Checks that every rated item belongs to the content catalog.

Compilation flags:

```
static
```

Template:

```
check_rated_catalog(Ratings,Contents)
```

Mode and number of proofs:

```
check_rated_catalog(+list(compound),+list(pair)) - one_or_error
```

Exceptions:

A rated item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

item_content_dataset_protocol, *recommender_common*

object

1.276.4 `jaccard_similarity`

Jaccard similarity of the observed key sets of two sparse vectors: intersection size divided by union size. Values, including zeros, are ignored. Each key must occur at most once per vector. Results range from 0.0 to 1.0; an empty union gives 0.0.

Availability:
`logtalk_load(recommender_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *similarity_metric_protocol*
Imports:
`public` *similarity_metric_common*
Uses:
list

Remarks:
(none)

Inherited public predicates:
similarity/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.276.5 `msd_similarity`

Inverse mean-squared-difference similarity over common keys: 1 / (1 + MSD). Results range from 0.0 to 1.0, with 1.0 for identical common ratings and 0.0 when no keys are common. Scaled differences avoid squaring original magnitudes. Each key must occur at most once per vector.

Availability:
`logtalk_load(recommender_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *similarity_metric_protocol*
Imports:
`public` *similarity_metric_common*
Uses:
list
numberlist

Remarks:
(none)

Inherited public predicates:
similarity/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.276.6 `pearson_similarity`

Pearson correlation between two sparse Key-Value vectors, computed only over common keys using shifted, scaled centering and unit normalization. Results range from -1.0 to 1.0. Fewer than two common keys, or exactly constant values in either common-key subset, gives 0.0.

Availability:
`logtalk_load(recommender_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *similarity_metric_protocol*
Imports:
`public` *similarity_metric_common*
Uses:
list

Remarks:
(none)

Inherited public predicates:
similarity/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.276.7 `rating_dataset_protocol`

Protocol for datasets used with collaborative filtering recommendation algorithms.

Availability:
`logtalk_load(recommender_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-02

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *rating/3*
  - *rating_count/1*
  - *rating_scale/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

rating/3

Enumerates by backtracking the known ratings. User and Item must be instantiated atomic identifiers and Rating is the numeric rating given by User to Item. At most one rating is declared per User-Item pair.

Compilation flags:

```
static
```

Template:

```
rating(User,Item,Rating)
```

Mode and number of proofs:

```
rating(-atomic,-atomic,-number) - zero_or_more
```

---

rating_count/1

Returns the number of known ratings. The declared count must match the number of ratings enumerated by rating/3.

Compilation flags:

```
static
```

Template:

```
rating_count(Count)
```

Mode and number of proofs:

```
rating_count(-positive_integer) - one
```

---

rating_scale/2

Returns the inclusive lower and upper bounds of the rating scale (e.g. 1 and 5). Fails when the dataset declares no fixed scale.

Compilation flags:

```
static
```

Template:

```
rating_scale(Min,Max)
```

Mode and number of proofs:

```
rating_scale(-number,-number) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

category

1.276.8 `recommender_common`

Shared predicates for recommender diagnostics, rating dataset validation, rating-matrix utilities, similarity metrics, and top-k retrieval.

Availability:

`logtalk_load(recommender_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:

`static`

Implements:

`public` *recommender_protocol*

Extends:

`public` *options*

Uses:

cosine_similarity
format
jaccard_similarity
list
msd_similarity
numberlist
pairs
pearson_similarity
spearman_similarity
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_recommender/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_recommender/1* *recommend/4* *recommender_options/2* *score/4* *valid_option/1* *valid_options/1* *valid_recommender/1*

- *Public predicates*
- *Protected predicates*
  - *recommender_valid_data/1*
  - *recommender_diagnostics_data/2*
  - *recommender_export_template/4*
  - *recommender_term_template/2*
  - *print_recommender_template/1*
  - *dataset_ratings/2*
  - *check_no_duplicate_ratings/1*
  - *check_ratings/2*
  - *check_query_identifiers/2*
  - *dataset_rating_scale/2*
  - *fallback_rating/5*
  - *clip_rating/3*
  - *recommend_from_ratings/5*
  - *users/2*
  - *items/2*
  - *user_vector/3*
  - *item_vector/3*
  - *global_mean_rating/2*
  - *user_mean_rating/3*
  - *item_mean_rating/3*
  - *cosine_similarity/3*
  - *pearson_similarity/3*
  - *jaccard_similarity/3*
  - *msd_similarity/3*
  - *spearman_similarity/3*
  - *top_k/3*
  - *base_recommender_diagnostics/5*
  - *valid_recommender_metadata/2*
  - *valid_recommender_metadata/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

recommender_valid_data/1

Required hook that validates implementation-specific model data and its consistency with diagnostics without instantiating the recommender.

Compilation flags:

```
static
```

Template:

```
recommender_valid_data(Recommender)
```

Mode and number of proofs:

```
recommender_valid_data(@compound) - zero_or_one
```

---

recommender_diagnostics_data/2

Hook predicate that importing recommender implementations must define in order to expose diagnostics metadata. A default implementation is provided that assumes the diagnostics list is the last argument of the recommender term; concrete implementations following that convention do not need to override it.

Compilation flags:

```
static
```

Template:

```
recommender_diagnostics_data(Recommender,Diagnostics)
```

Mode and number of proofs:

```
recommender_diagnostics_data(+compound,-list(compound)) - one
```

---

recommender_export_template/4

Hook predicate that importing recommender implementations must define in order to expose the exported recommender template for a given functor.

Compilation flags:

```
static
```

Template:

```
recommender_export_template(Dataset,Recommender,Functor,Template)
```

Mode and number of proofs:

```
recommender_export_template(+object_identifier,+compound,+atom,-callable) - one
```

---

recommender_term_template/2

Hook predicate that importing recommender implementations must define in order to expose the learned recommender term template used by pretty-printing helpers.

Compilation flags:

```
static
```

Template:

```
recommender_term_template(Recommender,Template)
```

Mode and number of proofs:

```
recommender_term_template(+compound,-callable) - one
```

---

print_recommender_template/1

Pretty-printing helper predicate used by importing recommender implementations to show the learned recommender term template.

Compilation flags:

```
static
```

Template:

```
print_recommender_template(Recommender)
```

Mode and number of proofs:

```
print_recommender_template(+compound) - one
```

---

dataset_ratings/2

Collects ratings with atomic user and item identifiers as `rating(User, Item, Rating)` terms. Checks for duplicate user-item pairs and a declared rating count matching the observed count.

Compilation flags:

```
static
```

Template:

```
dataset_ratings(Dataset,Ratings)
```

Mode and number of proofs:

```
dataset_ratings(+object_identifier,-list(compound)) - one_or_error
```

Exceptions:

The dataset contains no ratings:

```
domain_error(non_empty_ratings,Dataset)
```

A user or item identifier is a variable:

```
instantiation_error
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

The same `User-Item` pair is rated more than once:
`domain_error(duplicate_rating,User-Item)`
The declared rating count is a variable:
`instantiation_error`
The declared rating count is neither a variable nor an integer:
`type_error(integer,DeclaredCount)`
The declared rating count is an integer but is not positive:
`domain_error(positive_integer,DeclaredCount)`
The declared and observed rating counts differ:
`consistency_error(rating_count,DeclaredCount,ObservedCount)`

---

check_no_duplicate_ratings/1

Checks that ratings with atomic user and item identifiers contain no duplicate user-item pair.

Compilation flags:
`static`

Template:
`check_no_duplicate_ratings(Ratings)`
Mode and number of proofs:
`check_no_duplicate_ratings(+list(compound))` - `one_or_error`

Exceptions:
The same user-item pair is rated more than once:
`domain_error(duplicate_rating,User-Item)`

---

check_ratings/2

Checks that ratings are non-empty, have atomic user and item identifiers and numeric values, and fall within any declared `rating_scale/2`.

Compilation flags:
`static`

Template:
`check_ratings(Dataset,Ratings)`
Mode and number of proofs:
`check_ratings(+object_identifier,+list(compound))` - `one_or_error`

Exceptions:
`Ratings` is empty:
`domain_error(non_empty_ratings,Dataset)`
A user or item identifier is a variable:
`instantiation_error`

A user or item identifier is not atomic:
`type_error(atomic,Identifier)`
A rating value is not a number:
`type_error(number,Rating)`
A scale bound is not numeric:
`type_error(number,Bound)`
Scale bounds are reversed:
`domain_error(rating_scale,Min-Max)`
A rating value is a number but falls outside the declared rating scale:
`domain_error(rating_scale(Min,Max),Rating)`

---

check_query_identifiers/2

Checks that prediction identifiers are instantiated atomic terms.

Compilation flags:
`static`

Template:
`check_query_identifiers(User,Item)`
Mode and number of proofs:
`check_query_identifiers(@term,@term)` - `one_or_error`

Exceptions:
An identifier is a variable:
`instantiation_error`
An identifier is not atomic:
`type_error(atomic,Identifier)`

---

dataset_rating_scale/2

Returns `none` or a validated inclusive `scale(Min, Max)`.

Compilation flags:
`static`

Template:
`dataset_rating_scale(Dataset,Scale)`
Mode and number of proofs:
`dataset_rating_scale(+object_identifier,-term)` - `one_or_error`

Exceptions:
A scale bound is a variable:
`instantiation_error`

A scale bound is not numeric:

```
type_error(number,Bound)
```

Scale bounds are reversed:

```
domain_error(rating_scale,Min-Max)
```

---

fallback_rating/5

Returns the user mean, otherwise the item mean, otherwise the supplied global mean.

Compilation flags:

```
static
```

Template:

```
fallback_rating(Ratings,GlobalMean,User,Item,Rating)
```

Mode and number of proofs:

```
fallback_rating(+list(compound),+number,+atomic,+atomic,-number) - one
```

---

clip_rating/3

Clips a rating to a validated `scale(Min, Max)`; `none` leaves it unchanged.

Compilation flags:

```
static
```

Template:

```
clip_rating(Scale,Rating,Clipped)
```

Mode and number of proofs:

```
clip_rating(+term,+number,-number) - one
```

---

recommend_from_ratings/5

Scores unrated catalog items using self `score/4` and returns up to `N` descending-score pairs; ties use descending standard item order.

Compilation flags:

```
static
```

Template:

```
recommend_from_ratings(Recommender,Ratings,User,N,Recommendations)
```

Mode and number of proofs:

```
recommend_from_ratings(+compound,+list(compound),+atomic,+positive_integer,
-list(pair)) - one_or_error
```

Exceptions:

A required argument is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

User is not atomic:

```
type_error(atomic,User)
```

N is not an integer:

```
type_error(integer,N)
```

N is not positive:

```
domain_error(positive_integer,N)
```

---

users/2

Returns the sorted list of distinct users appearing in a list of `rating/3` terms.

Compilation flags:

```
static
```

Template:

```
users(Ratings,Users)
```

Mode and number of proofs:

```
users(+list(compound),-list(atomic)) - one
```

---

items/2

Returns the sorted list of distinct items appearing in a list of `rating/3` terms.

Compilation flags:

```
static
```

Template:

```
items(Ratings,Items)
```

Mode and number of proofs:

```
items(+list(compound),-list(atomic)) - one
```

---

user_vector/3

Returns the sparse rating vector of a user, as a list of `Item-Rating` pairs, from a list of `rating/3` terms. The vector is empty when the user has no ratings.

Compilation flags:

`static`

Template:

```
user_vector(Ratings,User,Vector)
```

Mode and number of proofs:

```
user_vector(+list(compound),+atomic,-list(pair)) - one
```

---

item_vector/3

Returns the sparse rating vector of an item, as a list of `User-Rating` pairs, from a list of `rating/3` terms. The vector is empty when the item has no ratings.

Compilation flags:

`static`

Template:

```
item_vector(Ratings,Item,Vector)
```

Mode and number of proofs:

```
item_vector(+list(compound),+atomic,-list(pair)) - one
```

---

global_mean_rating/2

Computes the mean of every rating in a list of `rating/3` terms (the global mean baseline).

Compilation flags:

`static`

Template:

```
global_mean_rating(Ratings,Mean)
```

Mode and number of proofs:

```
global_mean_rating(+list(compound),-float) - one_or_error
```

Exceptions:

`Ratings` is empty:

```
evaluation_error(zero_divisor)
```

---

user_mean_rating/3

Computes the mean of the ratings given by a user (the user mean baseline).

Compilation flags:
`static`

Template:
`user_mean_rating(Ratings,User,Mean)`
Mode and number of proofs:
`user_mean_rating(+list(compound),+atomic,-float)` - `one_or_error`

Exceptions:
`User` has no ratings:
`evaluation_error(zero_divisor)`

---

item_mean_rating/3

Computes the mean of the ratings received by an item (the item mean baseline).

Compilation flags:
`static`

Template:
`item_mean_rating(Ratings,Item,Mean)`
Mode and number of proofs:
`item_mean_rating(+list(compound),+atomic,-float)` - `one_or_error`

Exceptions:
`Item` has no ratings:
`evaluation_error(zero_divisor)`

---

cosine_similarity/3

Convenience predicate equivalent to `cosine_similarity::similarity/3`.

Compilation flags:
`static`

Template:
`cosine_similarity(Vector1,Vector2,Similarity)`
Mode and number of proofs:
`cosine_similarity(+list(pair),+list(pair),-number)` - `one`

---

pearson_similarity/3

Convenience predicate equivalent to `pearson_similarity::similarity/3`.

Compilation flags:
`static`

Template:
`pearson_similarity(Vector1,Vector2,Similarity)`
Mode and number of proofs:
`pearson_similarity(+list(pair),+list(pair),-number)` - `one`

---

jaccard_similarity/3

Convenience predicate equivalent to `jaccard_similarity::similarity/3`.

Compilation flags:
`static`

Template:
`jaccard_similarity(Vector1,Vector2,Similarity)`
Mode and number of proofs:
`jaccard_similarity(+list(pair),+list(pair),-number)` - `one`

---

msd_similarity/3

Convenience predicate equivalent to `msd_similarity::similarity/3`.

Compilation flags:
`static`

Template:
`msd_similarity(Vector1,Vector2,Similarity)`
Mode and number of proofs:
`msd_similarity(+list(pair),+list(pair),-number)` - `one`

---

spearman_similarity/3

Convenience predicate equivalent to `spearman_similarity::similarity/3`.

Compilation flags:
`static`

Template:
`spearman_similarity(Vector1,Vector2,Similarity)`
Mode and number of proofs:
`spearman_similarity(+list(pair),+list(pair),-number)` - `one`

---

top_k/3

Returns the `K` highest-scoring `Key-Score` pairs, sorted by decreasing score. Returns every pair, still sorted, when fewer than `K` are given; does not fail or throw an exception in that case.

Compilation flags:
`static`

Template:
`top_k(Pairs,K,TopK)`
Mode and number of proofs:
`top_k(+list(pair),+positive_integer,-list(pair))` - `one`

---

base_recommender_diagnostics/5

Builds the common part of a recommender diagnostics metadata list, combined with implementation-specific extra diagnostics terms.

Compilation flags:
`static`

Template:
`base_recommender_diagnostics(Model,RatingCount,Options,ExtraDiagnostics,`
`Diagnostics)`
Mode and number of proofs:
`base_recommender_diagnostics(+atom,+positive_integer,+list(compound),`
`+list(compound),-list(compound))` - `one`

---

valid_recommender_metadata/2

True when diagnostics metadata contains the expected model term, without instantiating the metadata.

Compilation flags:
`static`

Template:
`valid_recommender_metadata(Model,Diagnostics)`
Mode and number of proofs:
`valid_recommender_metadata(+atom,+list(compound))` - `zero_or_one`

---

valid_recommender_metadata/3

True when diagnostics metadata contains the expected model term and effective options, without instantiating the metadata.

Compilation flags:
`static`

Template:
`valid_recommender_metadata(Model,Options,Diagnostics)`
Mode and number of proofs:
`valid_recommender_metadata(+atom,+list(compound),+list(compound))` - `zero_or_one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.276.9 recommender_protocol

Protocol for recommendation models and user-item relevance scoring.

Availability:
`logtalk_load(recommender_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-04

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/3*
  - *learn/2*
  - *score/4*
  - *recommend/4*
  - *check_recommender/1*
  - *valid_recommender/1*
  - *diagnostics/2*
  - *diagnostic/2*
  - *recommender_options/2*
  - *export_to_clauses/4*
  - *export_to_file/4*
  - *print_recommender/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/3

Learns a recommender model from the given rating dataset object using the specified options.

Compilation flags:
`static`

Template:
`learn(Dataset,Recommender,Options)`
Mode and number of proofs:
`learn(+object_identifier,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

The dataset contains no ratings:

```
domain_error(non_empty_ratings,Dataset)
```

A user or item identifier is a variable:

```
instantiation_error
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

The same `User-Item` pair is rated more than once:

```
domain_error(duplicate_rating,User-Item)
```

The declared rating count is a variable:

```
instantiation_error
```

The declared rating count is neither a variable nor an integer:

```
type_error(integer,DeclaredCount)
```

The declared rating count is an integer but is not positive:

```
domain_error(positive_integer,DeclaredCount)
```

The declared and observed rating counts differ:

```
consistency_error(rating_count,DeclaredCount,ObservedCount)
```

A rating value is not a number:

```
type_error(number,Rating)
```

A scale bound is a variable:

```
instantiation_error
```

A scale bound is not numeric:

```
type_error(number,Bound)
```

Scale bounds are reversed:

```
domain_error(rating_scale,Min-Max)
```

A rating value falls outside the declared rating scale:

```
domain_error(rating_scale(Min,Max),Rating)
```

A content-based dataset has no catalog items:

```
domain_error(non_empty_catalog,Dataset)
```

A catalog or content item is declared more than once:

```
domain_error(duplicate_item,Item)
```

Content declarations do not cover the catalog exactly once:

```
domain_error(item_content_coverage,Dataset)
```

A rated item is absent from the content catalog:

```
domain_error(catalog_item,Item)
```

A content descriptor is invalid:

```
domain_error(item_content,Content)
```

Content representations are mixed:

```
domain_error(content_representation,Content)
```

A content list is not a proper list:

```
type_error(list,List)
```

A content descriptor, list, feature, or weight is a variable:

```
instantiation_error
```

A vector entry is not a pair:

```
type_error(pair,Entry)
```

A sparse vector repeats a feature key:

`domain_error(duplicate_feature,Feature)`

A sparse weight is not a number:

`type_error(number,Weight)`

A sparse weight is negative or nonfinite:

`domain_error(non_negative_finite_weight,Weight)`

A vector weight is not binary for an implementation requiring binary features:

`domain_error(binary_weight,Weight)`

A nonzero vector count is not an integer for an implementation requiring raw term counts:

`type_error(integer,Count)`

A content-based rating is not finite:

`domain_error(finite_rating,Rating)`

A selected rating is not positive in rating-weighted profile construction:

`domain_error(positive_rating_weight,Rating)`

Feature fitting produces no vocabulary:

`domain_error(non_empty_vocabulary,Corpus)`

`learn/2`

Learns a recommender model from the given rating dataset object using default options.

Compilation flags:

`static`

Template:

`learn(Dataset,Recommender)`

Mode and number of proofs:

`learn(+object_identifier,-compound)` - `one_or_error`

Exceptions:

The dataset contains no ratings:

`domain_error(non_empty_ratings,Dataset)`

A user or item identifier is a variable:

`instantiation_error`

A user or item identifier is not atomic:

`type_error(atomic,Identifier)`

The same `User-Item` pair is rated more than once:

`domain_error(duplicate_rating,User-Item)`

The declared rating count is a variable:

`instantiation_error`

The declared rating count is neither a variable nor an integer:

`type_error(integer,DeclaredCount)`

The declared rating count is an integer but is not positive:

`domain_error(positive_integer,DeclaredCount)`

The declared and observed rating counts differ:

`consistency_error(rating_count,DeclaredCount,ObservedCount)`

A rating value is not a number:

`type_error(number,Rating)`

A scale bound is a variable:

```
instantiation_error
```

A scale bound is not numeric:

```
type_error(number,Bound)
```

Scale bounds are reversed:

```
domain_error(rating_scale,Min-Max)
```

A rating value falls outside the declared rating scale:

```
domain_error(rating_scale(Min,Max),Rating)
```

A content-based dataset has no catalog items:

```
domain_error(non_empty_catalog,Dataset)
```

A catalog or content item is declared more than once:

```
domain_error(duplicate_item,Item)
```

Content declarations do not cover the catalog exactly once:

```
domain_error(item_content_coverage,Dataset)
```

A rated item is absent from the content catalog:

```
domain_error(catalog_item,Item)
```

A content descriptor is invalid:

```
domain_error(item_content,Content)
```

Content representations are mixed:

```
domain_error(content_representation,Content)
```

A content list is not a proper list:

```
type_error(list,List)
```

A content descriptor, list, feature, or weight is a variable:

```
instantiation_error
```

A vector entry is not a pair:

```
type_error(pair,Entry)
```

A sparse vector repeats a feature key:

```
domain_error(duplicate_feature,Feature)
```

A sparse weight is not a number:

```
type_error(number,Weight)
```

A sparse weight is negative or nonfinite:

```
domain_error(non_negative_finite_weight,Weight)
```

A vector weight is not binary for an implementation requiring binary features:

```
domain_error(binary_weight,Weight)
```

A nonzero vector count is not an integer for an implementation requiring raw term counts:

```
type_error(integer,Count)
```

A content-based rating is not finite:

```
domain_error(finite_rating,Rating)
```

A selected rating is not positive in rating-weighted profile construction:

```
domain_error(positive_rating_weight,Rating)
```

Feature fitting produces no vocabulary:

```
domain_error(non_empty_vocabulary,Corpus)
```

score/4

Scores the relevance of `Item` to `User` using the learned recommender. Higher scores indicate greater relevance. Scores may be predicted ratings or other relevance measures; their ranges and handling of unknown identifiers depend on the implementation.

Compilation flags:

`static`

Template:

`score(Recommender,User,Item,Score)`

Mode and number of proofs:

`score(+compound,+atomic,+atomic,-number)` - `one_or_error`

Exceptions:

`Recommender` is a variable:

`instantiation_error`

`Recommender` is not a valid recommender:

`domain_error(recommender,Recommender)`

A user or item identifier is a variable:

`instantiation_error`

A user or item identifier is not atomic:

`type_error(atomic,Identifier)`

`Item` is unknown to an implementation requiring catalog membership:

`domain_error(catalog_item,Item)`

---

recommend/4

Recommends up to `N` items to `User` using the learned recommender model, as a list of `Item-Score` pairs sorted by decreasing score. Whether items already rated by `User` are excluded, and how ties and an unknown `User` are handled, depends on the implementation.

Compilation flags:

`static`

Template:

`recommend(Recommender,User,N,Recommendations)`

Mode and number of proofs:

`recommend(+compound,+atomic,+positive_integer,-list(pair))` - `one_or_error`

Exceptions:

`Recommender` is a variable:

`instantiation_error`

`Recommender` is not a valid recommender:

`domain_error(recommender,Recommender)`

`N` is a variable:

`instantiation_error`

`N` is neither a variable nor an integer:

`type_error(integer,N)`

`N` is an integer but is not positive:
```
domain_error(positive_integer,N)
```
`User` is a variable:
```
instantiation_error
```
`User` is not atomic:
```
type_error(atomic,User)
```

---

check_recommender/1

Checks model data and diagnostics for the receiving implementation without instantiating the recommender. Throws an exception when the term is not a valid recommender representation.

Compilation flags:
```
static
```

Template:
```
check_recommender(Recommender)
```
Mode and number of proofs:
```
check_recommender(@compound) - one_or_error
```

Exceptions:
`Recommender` is a variable:
```
instantiation_error
```
`Recommender` is neither a variable nor a valid recommender:
```
domain_error(recommender,Recommender)
```

---

valid_recommender/1

True when a learned recommender term is structurally valid for the receiving implementation. Succeeds iff `check_recommender/1` succeeds without throwing an exception.

Compilation flags:
```
static
```

Template:
```
valid_recommender(Recommender)
```
Mode and number of proofs:
```
valid_recommender(@compound) - zero_or_one
```

---

diagnostics/2

Returns diagnostics metadata for a learned recommender.

Compilation flags:
```
static
```

Template:
```
diagnostics(Recommender,Diagnostics)
```
Mode and number of proofs:
```
diagnostics(+compound,-list(compound)) - one
```

---

diagnostic/2

Enumerates individual diagnostics metadata terms for a learned recommender.

Compilation flags:
```
static
```

Template:
```
diagnostic(Recommender,Diagnostic)
```
Mode and number of proofs:
```
diagnostic(+compound,?compound) - zero_or_more
```

---

recommender_options/2

Returns the effective training options recorded in a learned recommender diagnostics metadata.

Compilation flags:
```
static
```

Template:
```
recommender_options(Recommender,Options)
```
Mode and number of proofs:
```
recommender_options(+compound,-list(compound)) - one
```

---

export_to_clauses/4

Converts a recommender into a list of predicate clauses. `Functor` is the functor for the generated predicate clauses. When exporting a serialized recommender term, a noun such as `recommender` or `model` is usually clearer than a verb such as `recommend`.

Compilation flags:

```
static
```

Template:

```
export_to_clauses(Dataset,Recommender,Functor,Clauses)
```

Mode and number of proofs:

```
export_to_clauses(+object_identifier,+compound,+atom,-list(clause)) - one
```

---

export_to_file/4

Exports a recommender to a file. `Functor` is the functor for the generated predicate clauses. When exporting a serialized recommender term, a noun such as `recommender` or `model` is usually clearer than a verb such as `recommend`.

Compilation flags:

```
static
```

Template:

```
export_to_file(Dataset,Recommender,Functor,File)
```

Mode and number of proofs:

```
export_to_file(+object_identifier,+compound,+atom,+atom) - one
```

---

print_recommender/1

Prints a recommender to the current output stream in a human-readable format.

Compilation flags:

```
static
```

Template:

```
print_recommender(Recommender)
```

Mode and number of proofs:

```
print_recommender(+compound) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

rating_dataset_protocol, *similarity_metric_protocol*

category

1.276.10 `similarity_metric_common`

Shared helper predicates for sparse-vector overlap, numeric scaling, normalization, and correlation, reused by the library similarity metric objects.

Availability:

```
logtalk_load(recommender_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:

```
static
```

Uses:

list
numberlist

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
- *Protected predicates*
  - *common_pairs/3*
  - *vector_norm/2*
  - *scale_values/2*
  - *normalize_values/2*

      - *normalize_vector/2*
      - *bounded_similarity/2*
      - *centered_normalized_values/2*
      - *split_pairs/3*
      - *dot_product/3*
      - *sum_of_squares_list/2*
      - *mean_values/2*
      - *center/3*
  - *Private predicates*
  - *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

common_pairs/3

Collects the `Value1-Value2` pairs for the keys present in both sparse vectors. Assumes each vector declares at most one value per key.

Compilation flags:
`static`

Template:
`common_pairs(Vector1,Vector2,Pairs)`
Mode and number of proofs:
`common_pairs(+list(pair),+list(pair),-list(pair))` - `one`

---

vector_norm/2

Computes the Euclidean norm of the values of a sparse Key-Value vector.

Compilation flags:
`static`

Template:
`vector_norm(Vector,Norm)`
Mode and number of proofs:
`vector_norm(+list(pair),-float)` - `one`

---

scale_values/2

Scales numeric values by their maximum absolute magnitude, leaving empty and all-zero lists unchanged.

Compilation flags:
```
static
```

Template:
```
scale_values(Values,Scaled)
```
Mode and number of proofs:
```
scale_values(+list(number),-list(number)) - one
```

---

normalize_values/2

Unit-normalizes numeric values using bounded intermediate magnitudes, leaving empty and all-zero lists unchanged.

Compilation flags:
```
static
```

Template:
```
normalize_values(Values,Normalized)
```
Mode and number of proofs:
```
normalize_values(+list(number),-list(number)) - one
```

---

normalize_vector/2

Unit-normalizes a sparse numeric vector while preserving its keys and their order.

Compilation flags:
```
static
```

Template:
```
normalize_vector(Vector,Normalized)
```
Mode and number of proofs:
```
normalize_vector(+list(pair),-list(pair)) - one
```

---

bounded_similarity/2

Bounds a computed cosine or correlation score to its mathematical range, removing endpoint roundoff.

Compilation flags:
`static`

Template:
`bounded_similarity(Score,Similarity)`
Mode and number of proofs:
`bounded_similarity(+number,-number)` - `one`

---

centered_normalized_values/2

Centers and unit-normalizes numeric values in shifted, scaled coordinates. Empty lists remain empty and exactly constant lists become zeros.

Compilation flags:
`static`

Template:
`centered_normalized_values(Values,Normalized)`
Mode and number of proofs:
`centered_normalized_values(+list(number),-list(number))` - `one`

---

split_pairs/3

Splits a list of `Value1-Value2` pairs into separate, position-aligned lists of first and second values.

Compilation flags:
`static`

Template:
`split_pairs(Pairs,Values1,Values2)`
Mode and number of proofs:
`split_pairs(+list(pair),-list(number),-list(number))` - `one`

---

dot_product/3

Computes the dot product of two position-aligned numeric lists of the same length.

Compilation flags:
`static`

Template:
`dot_product(Values1,Values2,Dot)`
Mode and number of proofs:
`dot_product(+list(number),+list(number),-number)` - `one`

---

sum_of_squares_list/2

Computes the sum of the squares of a numeric list.

Compilation flags:
`static`

Template:
`sum_of_squares_list(Values,SumSquares)`
Mode and number of proofs:
`sum_of_squares_list(+list(number),-number)` - `one`

---

mean_values/2

Computes the arithmetic mean of a non-empty numeric list.

Compilation flags:
`static`

Template:
`mean_values(Values,Mean)`
Mode and number of proofs:
`mean_values(+list(number),-float)` - `one_or_error`

Exceptions:
`Values` is empty:
`evaluation_error(zero_divisor)`

`center/3`

Subtracts a value (typically the mean) from every element of a numeric list.

Compilation flags:
`static`

Template:
`center(Values,Value,Centered)`
Mode and number of proofs:
`center(+list(number),+number,-list(number))` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.276.11 `similarity_metric_protocol`

Protocol for pluggable similarity metrics between two sparse vectors, used to compare two users (by their item ratings) or two items (by the ratings they received).

Availability:
`logtalk_load(recommender_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-02

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

  - *similarity/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

similarity/3

Computes a similarity score between two sparse vectors of `Key-Value` pairs with unique ground keys and finite numeric values, unless an implementation ignores values. Higher scores indicate greater similarity. The range and handling of empty or non-overlapping vectors depend on the implementation.

Compilation flags:
`static`

Template:
`similarity(Vector1,Vector2,Similarity)`
Mode and number of proofs:
`similarity(+list(pair),+list(pair),-number)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.276.12 spearman_similarity

Spearman rank correlation over common keys, using average ranks for numeric ties. Results range from -1.0 to 1.0. Fewer than two common keys or constant common values in either vector gives 0.0. Each key must occur at most once per vector.

Availability:
`logtalk_load(recommender_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *similarity_metric_protocol*
Imports:
`public` *similarity_metric_common*
Uses:
list

Remarks:
(none)

Inherited public predicates:
similarity/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.277 recorded_database

object

1.277.1 `recorded_database`

Legacy recorded database predicates. Provides an application global database.

Availability:
`logtalk_load(recorded_database(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2023-12-17

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *recorded_database_core*

Remarks:
(none)

Inherited public predicates:
erase/1 *instance/2* *recorda/2* *recorda/3* *recorded/2* *recorded/3* *recordz/2* *recordz/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.277.2 recorded_database_core

Legacy recorded database predicates. Can be imported into an object to provide a local database.

Availability:
`logtalk_load(recorded_database(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2023-12-17

Compilation flags:
`static`

Dependencies:

(none)

Remarks:

- References: Opaque ground terms.

Inherited public predicates:
(none)

- *Public predicates*
  - *recorda/3*
  - *recorda/2*
  - *recordz/3*
  - *recordz/2*
  - *recorded/3*
  - *recorded/2*
  - *erase/1*
  - *instance/2*
- *Protected predicates*
- *Private predicates*
  - *record_/3*
  - *reference_/1*
- *Operators*

Public predicates

recorda/3

Adds a term as the first term for the given key, returning its reference.

Compilation flags:
`static`

Template:
`recorda(Key,Term,Reference)`

Mode and number of proofs:
`recorda(+recorded_database_key,+term,--recorded_database_reference)` - `one_or_error`

Exceptions:
`Key` is a variable:
`instantiation_error`
`Key` is neither a variable nor an atomic term or compound term:
`type_error(recorded_database_key,Key)`
`Reference` is a not a variable:
`uninstantiation_error(Reference)`

recorda/2

Adds a term as the first term for the given key.

Compilation flags:
`static`

Template:
`recorda(Key,Term)`
Mode and number of proofs:
`recorda(+recorded_database_key,+term)` - `one_or_error`

Exceptions:
`Key` is a variable:
`instantiation_error`
`Key` is neither a variable nor an atomic term or compound term:
`type_error(recorded_database_key,Key)`

---

recordz/3

Adds a term as the last term for the given key, returning its reference.

Compilation flags:
`static`

Template:
`recordz(Key,Term,Reference)`
Mode and number of proofs:
`recordz(+recorded_database_key,+term,--recorded_database_reference)` - `one_or_error`

Exceptions:
`Key` is a variable:
`instantiation_error`
`Key` is neither a variable nor an atomic term or compound term:
`type_error(recorded_database_key,Key)`
`Reference` is a not a variable:
`uninstantiation_error(Reference)`

---

recordz/2

Adds a term as the last term for the given key.

Compilation flags:
`static`

Template:
`recordz(Key,Term)`
Mode and number of proofs:
`recordz(+recorded_database_key,+term)` - `one_or_error`

Exceptions:
`Key` is a variable:
`instantiation_error`
`Key` is neither a variable nor an atomic term or compound term:
`type_error(recorded_database_key,Key)`

---

recorded/3

Enumerates, by backtracking, all record key-term pairs and their references.

Compilation flags:
`static`

Template:
`recorded(Key,Term,Reference)`
Mode and number of proofs:
`recorded(?recorded_database_key,?term,-recorded_database_reference)` - `zero_or_more`
`recorded(?recorded_database_key,?term,+recorded_database_reference)` - `zero_or_one`

---

recorded/2

Enumerates, by backtracking, all record key-term pairs.

Compilation flags:
`static`

Template:
`recorded(Key,Term)`
Mode and number of proofs:
`recorded(?recorded_database_key,?term)` - `zero_or_more`

---

erase/1

Erases the record indexed by the given reference. Fails if there is no record with the given reference.

Compilation flags:
`static`

Template:
`erase(Reference)`
Mode and number of proofs:
`erase(@recorded_database_reference)` - `zero_or_one_or_error`

Exceptions:
`Reference` is a variable:
`instantiation_error`

---

instance/2

.

Compilation flags:
`static`

Template:
`instance(Reference,Term)`
Mode and number of proofs:
`instance(@recorded_database_reference,?term)` - `zero_or_one_or_error`

Exceptions:
`Reference` is a variable:
`instantiation_error`

Protected predicates

(none)

Private predicates

record_/3

Records table.

Compilation flags:
`dynamic`

Template:

```
record_(Key,Term,Reference)
```

Mode and number of proofs:

```
record_(?recorded_database_key,?term,?recorded_database_reference) - zero_or_more
```

---

reference_/1

Reference count.

Compilation flags:

```
dynamic
```

Template:

```
reference_(Reference)
```

Mode and number of proofs:

```
reference_(?non_negative_integer) - zero_or_one
```

Operators

(none)

1.278 redis

object

1.278.1 redis

Redis client library with support for strings, keys, hashes, lists, sets, and sorted sets. Inspired by Sean Charles GNU Prolog Redis client.

Availability:

```
logtalk_load(redis(loader))
```

Author: Paulo Moura
Version: 0:7:0
Date: 2026-02-10

Compilation flags:

```
static, context_switching_calls
```

Provides:

logtalk::message_tokens//2

Uses:

list
logtalk
socket

Remarks:

- Command representation: Use the Redis command name as the functor of a compound term where the arguments are the command arguments.
- Valid arguments: Atoms, integers, and floats. Always use atoms instead of double-quoted "strings". This helps portability by not depending on the value of the `double_quotes` flag.
- Wrapper predicates: The library provides convenient wrapper predicates for common Redis operations. For operations without a wrapper, use the generic `send/3` predicate.

Inherited public predicates:

(none)

- *Public predicates*
  - *connect/1*
  - *connect/3*
  - *disconnect/1*
  - *send/3*
  - *console/1*
  - *get/3*
  - *set/4*
  - *append/4*
  - *getrange/5*
  - *setrange/5*
  - *strlen/3*
  - *mget/3*
  - *mset/3*
  - *incr/3*
  - *decr/3*
  - *incrby/4*
  - *decrby/4*
  - *del/3*
  - *exists/3*
  - *keys/3*
  - *ttl/3*
  - *expire/4*
  - *persist/3*
  - *rename/4*
  - *type/3*
  - *hset/5*
  - *hget/4*
  - *hgetall/3*
  - *hdel/4*
  - *hexists/4*
  - *hkeys/3*
  - *hvals/3*
  - *hlen/3*
  - *lpush/4*

      - *rpush/4*
      - *lpop/3*
      - *rpop/3*
      - *lrange/5*
      - *llen/3*
      - *lrem/5*
      - *ltrim/5*
      - *sadd/4*
      - *srem/4*
      - *smembers/3*
      - *sismember/4*
      - *scard/3*
      - *zadd/5*
      - *zrem/4*
      - *zrange/5*
      - *zrank/4*
      - *zcard/3*
      - *zscore/4*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

connect/1

Connect to a Redis server running on localhost using the default 6379 port.

Compilation flags:
```
static
```

Template:
```
connect(Connection)
```
Mode and number of proofs:
```
connect(--ground) - one
```

---

connect/3

Connect to a Redis server running on the given host and port.

Compilation flags:
```
static
```

Template:
```
connect(Host,Port,Connection)
```
Mode and number of proofs:

```
connect(+atom,+integer,--ground) - one
```

---

disconnect/1

Disconnect from a Redis server.

Compilation flags:

```
static
```

Template:

```
disconnect(Connection)
```

Mode and number of proofs:

```
disconnect(++ground) - one
```

---

send/3

Sends a request to the a Redis server and returns its reply.

Compilation flags:

```
static
```

Template:

```
send(Connection,Request,Reply)
```

Mode and number of proofs:

```
send(++ground,++callable,--callable) - one
```

---

console/1

Sends a request to a Redis server running on localhost at the default 6379 port and prints the reply.

Compilation flags:

```
static
```

Template:

```
console(Request)
```

Mode and number of proofs:

```
console(++callable) - one
```

---

get/3

Gets the value of a key.

Compilation flags:
```
static
```

Template:
```
get(Connection,Key,Value)
```
Mode and number of proofs:
```
get(+ground,+atom,-atom) - one
```

---

set/4

Sets the value of a key.

Compilation flags:
```
static
```

Template:
```
set(Connection,Key,Value,Status)
```
Mode and number of proofs:
```
set(+ground,+atom,+ground,-atom) - one
```

---

append/4

Appends a value to a key. Returns the length of the string after the append.

Compilation flags:
```
static
```

Template:
```
append(Connection,Key,Value,Length)
```
Mode and number of proofs:
```
append(+ground,+atom,+ground,-integer) - one
```

---

getrange/5

Gets a substring of the string stored at a key.

Compilation flags:
```
static
```

Template:
```
getrange(Connection,Key,Start,End,Substring)
```
Mode and number of proofs:
```
getrange(+ground,+atom,+integer,+integer,-atom) - one
```

---

setrange/5

Overwrites part of a string at a key starting at the specified offset. Returns the length of the string after modification.

Compilation flags:
```
static
```

Template:
```
setrange(Connection,Key,Offset,Value,Length)
```
Mode and number of proofs:
```
setrange(+ground,+atom,+integer,+ground,-integer) - one
```

---

strlen/3

Gets the length of the value stored at a key.

Compilation flags:
```
static
```

Template:
```
strlen(Connection,Key,Length)
```
Mode and number of proofs:
```
strlen(+ground,+atom,-integer) - one
```

---

mget/3

Gets the values of multiple keys. Returns a list of values.

Compilation flags:
```
static
```

Template:
```
mget(Connection,Keys,Values)
```
Mode and number of proofs:
```
mget(+ground,+list(atom),-list) - one
```

---

mset/3

Sets multiple key-value pairs. Pairs should be provided as a flat list [Key1, Value1, Key2, Value2, ...].

Compilation flags:
```
static
```

Template:
```
mset(Connection,Pairs,Status)
```
Mode and number of proofs:
```
mset(+ground,+list,-atom) - one
```

---

incr/3

Increments the integer value of a key by one. Returns the value after increment.

Compilation flags:
```
static
```

Template:
```
incr(Connection,Key,Value)
```
Mode and number of proofs:
```
incr(+ground,+atom,-integer) - one
```

---

decr/3

Decrements the integer value of a key by one. Returns the value after decrement.

Compilation flags:
`static`

Template:
`decr(Connection,Key,Value)`
Mode and number of proofs:
`decr(+ground,+atom,-integer)` - `one`

---

incrby/4

Increments the integer value of a key by the specified amount. Returns the value after increment.

Compilation flags:
`static`

Template:
`incrby(Connection,Key,Increment,Value)`
Mode and number of proofs:
`incrby(+ground,+atom,+integer,-integer)` - `one`

---

decrby/4

Decrements the integer value of a key by the specified amount. Returns the value after decrement.

Compilation flags:
`static`

Template:
`decrby(Connection,Key,Decrement,Value)`
Mode and number of proofs:
`decrby(+ground,+atom,+integer,-integer)` - `one`

---

del/3

Deletes a key. Returns the number of keys removed.

Compilation flags:
```
static
```

Template:
```
del(Connection,Key,Count)
```
Mode and number of proofs:
```
del(+ground,+atom,-integer) - one
```

---

exists/3

Checks if a key exists. Returns 1 if the key exists, 0 otherwise.

Compilation flags:
```
static
```

Template:
```
exists(Connection,Key,Exists)
```
Mode and number of proofs:
```
exists(+ground,+atom,-integer) - one
```

---

keys/3

Finds all keys matching a pattern. Returns a list of keys.

Compilation flags:
```
static
```

Template:
```
keys(Connection,Pattern,Keys)
```
Mode and number of proofs:
```
keys(+ground,+atom,-list) - one
```

---

ttl/3

Gets the time to live for a key in seconds. Returns -1 if the key has no expiry, -2 if the key does not exist.

Compilation flags:
```
static
```

Template:
```
ttl(Connection,Key,Seconds)
```
Mode and number of proofs:
```
ttl(+ground,+atom,-integer) - one
```

---

expire/4

Sets a timeout on a key in seconds. Returns 1 if the timeout was set, 0 if the key does not exist.

Compilation flags:
```
static
```

Template:
```
expire(Connection,Key,Seconds,Result)
```
Mode and number of proofs:
```
expire(+ground,+atom,+integer,-integer) - one
```

---

persist/3

Removes the expiration from a key. Returns 1 if the timeout was removed, 0 if the key does not exist or has no timeout.

Compilation flags:
```
static
```

Template:
```
persist(Connection,Key,Result)
```
Mode and number of proofs:
```
persist(+ground,+atom,-integer) - one
```

---

rename/4

Renames a key. Returns status OK or error if the key does not exist.

Compilation flags:
```
static
```

Template:
```
rename(Connection,OldKey,NewKey,Status)
```
Mode and number of proofs:
```
rename(+ground,+atom,+atom,-atom) - one
```

---

type/3

Gets the type of the value stored at a key. Returns one of: string, list, set, zset, hash, stream, none.

Compilation flags:
```
static
```

Template:
```
type(Connection,Key,Type)
```
Mode and number of proofs:
```
type(+ground,+atom,-atom) - one
```

---

hset/5

Sets a field in a hash. Returns 1 if a new field was created, 0 if the field was updated.

Compilation flags:
```
static
```

Template:
```
hset(Connection,Key,Field,Value,Result)
```
Mode and number of proofs:
```
hset(+ground,+atom,+atom,+ground,-integer) - one
```

---

hget/4

Gets the value of a field in a hash.

Compilation flags:
`static`

Template:
`hget(Connection,Key,Field,Value)`
Mode and number of proofs:
`hget(+ground,+atom,+atom,-ground) - one`

---

hgetall/3

Gets all fields and values in a hash. Returns a flat list of alternating fields and values.

Compilation flags:
`static`

Template:
`hgetall(Connection,Key,FieldsValues)`
Mode and number of proofs:
`hgetall(+ground,+atom,-list) - one`

---

hdel/4

Deletes a field from a hash. Returns the number of fields removed.

Compilation flags:
`static`

Template:
`hdel(Connection,Key,Field,Count)`
Mode and number of proofs:
`hdel(+ground,+atom,+atom,-integer) - one`

---

hexists/4

Checks if a field exists in a hash. Returns 1 if the field exists, 0 otherwise.

Compilation flags:
```
static
```

Template:
```
hexists(Connection,Key,Field,Exists)
```
Mode and number of proofs:
```
hexists(+ground,+atom,+atom,-integer) - one
```

---

hkeys/3

Gets all field names in a hash.

Compilation flags:
```
static
```

Template:
```
hkeys(Connection,Key,Fields)
```
Mode and number of proofs:
```
hkeys(+ground,+atom,-list) - one
```

---

hvals/3

Gets all values in a hash.

Compilation flags:
```
static
```

Template:
```
hvals(Connection,Key,Values)
```
Mode and number of proofs:
```
hvals(+ground,+atom,-list) - one
```

---

hlen/3

Gets the number of fields in a hash.

Compilation flags:
`static`

Template:
`hlen(Connection,Key,Count)`
Mode and number of proofs:
`hlen(+ground,+atom,-integer)` - `one`

---

lpush/4

Prepends a value to a list. Returns the length of the list after the push.

Compilation flags:
`static`

Template:
`lpush(Connection,Key,Value,Length)`
Mode and number of proofs:
`lpush(+ground,+atom,+ground,-integer)` - `one`

---

rpush/4

Appends a value to a list. Returns the length of the list after the push.

Compilation flags:
`static`

Template:
`rpush(Connection,Key,Value,Length)`
Mode and number of proofs:
`rpush(+ground,+atom,+ground,-integer)` - `one`

---

lpop/3

Removes and returns the first element of a list.

Compilation flags:
```
static
```

Template:
```
lpop(Connection,Key,Value)
```
Mode and number of proofs:
```
lpop(+ground,+atom,-ground) - one
```

---

rpop/3

Removes and returns the last element of a list.

Compilation flags:
```
static
```

Template:
```
rpop(Connection,Key,Value)
```
Mode and number of proofs:
```
rpop(+ground,+atom,-ground) - one
```

---

lrange/5

Gets a range of elements from a list. Indices are zero-based.

Compilation flags:
```
static
```

Template:
```
lrange(Connection,Key,Start,Stop,Elements)
```
Mode and number of proofs:
```
lrange(+ground,+atom,+integer,+integer,-list) - one
```

---

llen/3

Gets the length of a list.

Compilation flags:

```
static
```

Template:

```
llen(Connection,Key,Length)
```

Mode and number of proofs:

```
llen(+ground,+atom,-integer) - one
```

---

lrem/5

Removes elements from a list. Count > 0: remove from head, Count < 0: remove from tail, Count = 0: remove all. Returns the number of removed elements.

Compilation flags:

```
static
```

Template:

```
lrem(Connection,Key,Count,Value,Removed)
```

Mode and number of proofs:

```
lrem(+ground,+atom,+integer,+ground,-integer) - one
```

---

ltrim/5

Trims a list to the specified range.

Compilation flags:

```
static
```

Template:

```
ltrim(Connection,Key,Start,Stop,Status)
```

Mode and number of proofs:

```
ltrim(+ground,+atom,+integer,+integer,-atom) - one
```

---

sadd/4

Adds a member to a set. Returns the number of elements added.

Compilation flags:
```
static
```

Template:
```
sadd(Connection,Key,Member,Count)
```
Mode and number of proofs:
```
sadd(+ground,+atom,+ground,-integer) - one
```

---

srem/4

Removes a member from a set. Returns the number of elements removed.

Compilation flags:
```
static
```

Template:
```
srem(Connection,Key,Member,Count)
```
Mode and number of proofs:
```
srem(+ground,+atom,+ground,-integer) - one
```

---

smembers/3

Gets all members in a set.

Compilation flags:
```
static
```

Template:
```
smembers(Connection,Key,Members)
```
Mode and number of proofs:
```
smembers(+ground,+atom,-list) - one
```

---

sismember/4

Checks if a value is a member of a set. Returns 1 if the member exists, 0 otherwise.

Compilation flags:
```
static
```

Template:
```
sismember(Connection,Key,Member,IsMember)
```
Mode and number of proofs:
```
sismember(+ground,+atom,+ground,-integer) - one
```

---

scard/3

Gets the number of members in a set.

Compilation flags:
```
static
```

Template:
```
scard(Connection,Key,Count)
```
Mode and number of proofs:
```
scard(+ground,+atom,-integer) - one
```

---

zadd/5

Adds a member with a score to a sorted set. Returns the number of elements added.

Compilation flags:
```
static
```

Template:
```
zadd(Connection,Key,Score,Member,Count)
```
Mode and number of proofs:
```
zadd(+ground,+atom,+number,+ground,-integer) - one
```

---

zrem/4

Removes a member from a sorted set. Returns the number of elements removed.

Compilation flags:

```
static
```

Template:

```
zrem(Connection,Key,Member,Count)
```

Mode and number of proofs:

```
zrem(+ground,+atom,+ground,-integer) - one
```

---

zrange/5

Gets a range of members from a sorted set, ordered from lowest to highest score.

Compilation flags:

```
static
```

Template:

```
zrange(Connection,Key,Start,Stop,Members)
```

Mode and number of proofs:

```
zrange(+ground,+atom,+integer,+integer,-list) - one
```

---

zrank/4

Gets the rank (index) of a member in a sorted set, ordered from lowest to highest score. Returns nil if the member does not exist.

Compilation flags:

```
static
```

Template:

```
zrank(Connection,Key,Member,Rank)
```

Mode and number of proofs:

```
zrank(+ground,+atom,+ground,-integer) - one
```

---

zcard/3

Gets the number of members in a sorted set.

Compilation flags:
```
static
```

Template:
```
zcard(Connection,Key,Count)
```
Mode and number of proofs:
```
zcard(+ground,+atom,-integer) - one
```

---

zscore/4

Gets the score of a member in a sorted set.

Compilation flags:
```
static
```

Template:
```
zscore(Connection,Key,Member,Score)
```
Mode and number of proofs:
```
zscore(+ground,+atom,+ground,-ground) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.279 regression_protocols

protocol

1.279.1 regression_dataset_protocol

Protocol for datasets used with regression algorithms.

Availability:
`logtalk_load(regression_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-21

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *attribute_values/2*
  - *target/1*
  - *example/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

attribute_values/2

Enumerates by backtracking the attributes and their possible values. For discrete attributes, `Values` is a list of possible values. For continuous (numeric) attributes, `Values` is the atom `continuous`.

Compilation flags:
`static`

Template:
`attribute_values(Attribute,Values)`
Mode and number of proofs:
`attribute_values(?atom,-list(atom))` - `zero_or_more`
`attribute_values(?atom,-atom)` - `zero_or_more`

target/1

Returns the name of the numeric target attribute.

Compilation flags:
`static`

Template:
`target(Target)`
Mode and number of proofs:
`target(-atom)` - `one`

---

example/3

Enumerates by backtracking the examples in the dataset. Each example has an `Id`, a numeric `Target` value, and a list of `Attribute-Value` pairs.

Compilation flags:
`static`

Template:
`example(Id,Target,AttributeValues)`
Mode and number of proofs:
`example(-integer,-number,-list(pair))` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.279.2 regressor_common

Shared predicates for regressor learning defaults, diagnostics, validation, dataset validation, export, and pretty-print helpers.

Availability:
`logtalk_load(regression_protocols(loader))`

Author: Paulo Moura

Version: 1:1:1
Date: 2026-06-14

Compilation flags:
`static`

Implements:
`public` *regressor_protocol*
Extends:
`public` *options*
Uses:
format
linear_algebra
list
numberlist
population
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
- *Protected predicates*
  - *regressor_diagnostics_data/2*
  - *regressor_export_template/4*
  - *regressor_term_template/2*
  - *dataset_attributes/2*
  - *dataset_examples/2*
  - *check_examples/2*
  - *print_regressor_template/1*
  - *base_regressor_diagnostics/6*
  - *valid_attribute_names/1*
  - *valid_attribute_declarations/1*
  - *valid_discrete_values/1*
  - *valid_regression_encoders/1*
  - *valid_regressor_options/1*
  - *valid_regressor_metadata/2*
  - *valid_diagnostic_count/3*
  - *valid_linear_model_diagnostics/1*
  - *valid_encoded_rows/2*
  - *encoded_feature_count/2*
  - *continuous_stats/5*

  - *examples_to_rows/3*
  - *encode_instance/3*
  - *fit_linear_model/7*
  - *fit_ridge_model/7*
  - *valid_feature_labels/1*
  - *valid_regression_tree/2*
- *Private predicates*
  - *check_attribute_declarations/1*
  - *known_attribute_values/3*
  - *check_attribute_bindings/2*
  - *check_declared_attribute_bindings/2*
  - *check_undeclared_attribute_bindings/2*
  - *attribute_occurrences/4*
  - *declared_attribute_names/2*
  - *encoder_attribute_names/2*
  - *encode_instance_checked/3*
  - *valid_linear_model_convergence/1*
  - *normalize_continuous/4*
  - *check_categorical_value/3*
  - *one_hot_encode/4*
  - *missing_one_hot_encode/3*
  - *zero_vector_from_values/3*
  - *build_linear_encoders/4*
  - *train_linear_model/6*
  - *rows_to_design_matrix/3*
  - *residual_sum_of_squares/4*
  - *train_ridge_model/6*
  - *ridge_feature_statistics/3*
  - *accumulate_feature_statistics/5*
  - *add_squared_vector/3*
  - *feature_penalty_profiles/5*
  - *active_feature_count/2*
  - *compress_rows/3*
  - *compress_features/3*
  - *compress_vector/3*
  - *build_ridge_system/5*
  - *accumulate_ridge_system/5*
  - *add_outer_product/4*
  - *regularize_ridge_matrix/4*
  - *solve_linear_system/4*
  - *augment_rows/3*
  - *triangularize/2*
  - *select_pivot_row/3*
  - *leading_magnitude/2*
  - *ensure_non_zero/1*
  - *eliminate_rows/5*
  - *scaled_row_difference/4*

- *back_substitution/2*
- *maximum_linear_system_residual/4*
- *expand_weights/3*

- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

regressor_diagnostics_data/2

Default hook predicate for exposing diagnostics metadata from a regressor term. Importing implementations may override it when using a non-standard regressor representation.

Compilation flags:

```
static
```

Template:

```
regressor_diagnostics_data(Regressor,Diagnostics)
```

Mode and number of proofs:

```
regressor_diagnostics_data(+compound,-list(compound)) - one
```

---

regressor_export_template/4

Hook predicate that importing regressor implementations must define in order to expose the exported regressor template for a given functor.

Compilation flags:

```
static
```

Template:

```
regressor_export_template(Dataset,Regressor,Functor,Template)
```

Mode and number of proofs:

```
regressor_export_template(+object_identifier,+compound,+atom,-callable) - one
```

---

regressor_term_template/2

Hook predicate that importing regressor implementations must define in order to expose the learned regressor term template used by pretty-printing helpers.

Compilation flags:

```
static
```

Template:

```
regressor_term_template(Regressor,Template)
```

Mode and number of proofs:

```
regressor_term_template(+compound,-callable) - one
```

---

dataset_attributes/2

Collects the dataset attribute declarations as *Attribute-Values* pairs.

Compilation flags:

```
static
```

Template:

```
dataset_attributes(Dataset,Attributes)
```

Mode and number of proofs:

```
dataset_attributes(+object_identifier,-list(pair)) - one
```

---

dataset_examples/2

Collects the dataset examples as *example(Id, TargetValue, AttributeValues)* terms.

Compilation flags:

```
static
```

Template:

```
dataset_examples(Dataset,Examples)
```

Mode and number of proofs:

```
dataset_examples(+object_identifier,-list(compound)) - one
```

---

check_examples/2

Validates that the collected examples list is non-empty, only contains numeric targets, and only uses declared attributes without duplicate bindings.

Compilation flags:

```
static
```

Template:

```
check_examples(Dataset,Examples)
```

Mode and number of proofs:

```
check_examples(+object_identifier,+list) - one_or_error
```

Exceptions:

Examples is the empty list:

```
domain_error(non_empty_dataset,Dataset)
```

An example target is not numeric:

```
type_error(number,Target)
```

An example repeats a declared attribute binding:

```
domain_error(attribute_occurrences,Attribute)
```

An example contains an undeclared attribute binding:

```
domain_error(declared_attribute,Attribute)
```

---

print_regressor_template/1

Pretty-printing helper predicate used by importing regressor implementations to show the learned regressor term template.

Compilation flags:

```
static
```

Template:

```
print_regressor_template(Regressor)
```

Mode and number of proofs:

```
print_regressor_template(+compound) - one
```

---

base_regressor_diagnostics/6

Builds common diagnostics metadata terms for a learned regressor and appends regressor-specific diagnostics terms.

Compilation flags:

```
static
```

Template:

```
base_regressor_diagnostics(Model,Target,TrainingExampleCount,Options,
ExtraDiagnostics,Diagnostics)
```

Mode and number of proofs:

```
base_regressor_diagnostics(+atom,+atom,+integer,+list(compound),+list(compound),
-list(compound)) - one
```

---

valid_attribute_names/1

True when a list of attribute names is a proper list of distinct atoms.

Compilation flags:

```
static
```

Template:

```
valid_attribute_names(AttributeNames)
```

Mode and number of proofs:

```
valid_attribute_names(+list(atom)) - zero_or_one
```

---

valid_attribute_declarations/1

True when a list of attribute declarations is a proper list of distinct `Attribute-Values` pairs where values are either `continuous` or a valid discrete value list.

Compilation flags:

```
static
```

Template:

```
valid_attribute_declarations(Attributes)
```

Mode and number of proofs:

```
valid_attribute_declarations(+list(pair)) - zero_or_one
```

---

valid_discrete_values/1

True when a list of categorical values is non-empty, contains only nonvar terms, and has no duplicates.

Compilation flags:

```
static
```

Template:

```
valid_discrete_values(Values)
```

Mode and number of proofs:

```
valid_discrete_values(+list) - zero_or_one
```

---

valid_regression_encoders/1

True when a list of encoders only contains valid `continuous/3` or `categorical/2` encoder terms with distinct attributes.

Compilation flags:
`static`

Template:
`valid_regression_encoders(Encoders)`
Mode and number of proofs:
`valid_regression_encoders(+list(compound))` - `zero_or_one`

---

valid_regressor_options/1

True when a list of options is structurally valid for the receiving regressor implementation.

Compilation flags:
`static`

Template:
`valid_regressor_options(Options)`
Mode and number of proofs:
`valid_regressor_options(+list(compound))` - `zero_or_one`

---

valid_regressor_metadata/2

True when diagnostics metadata contains the expected model term and records a structurally valid effective options list.

Compilation flags:
`static`

Template:
`valid_regressor_metadata(Model,Diagnostics)`
Mode and number of proofs:
`valid_regressor_metadata(+atom,+list(compound))` - `zero_or_one`

---

valid_diagnostic_count/3

True when diagnostics contains a count term with the given functor and integer value.

Compilation flags:
`static`

Template:
`valid_diagnostic_count(Functor,Diagnostics,Count)`
Mode and number of proofs:
`valid_diagnostic_count(+atom,+list(compound),+integer)` - `zero_or_one`

---

valid_linear_model_diagnostics/1

True when diagnostics contains structurally valid linear-model optimization metadata terms for convergence, completed iterations, and final parameter delta.

Compilation flags:
`static`

Template:
`valid_linear_model_diagnostics(Diagnostics)`
Mode and number of proofs:
`valid_linear_model_diagnostics(+list(compound))` - `zero_or_one`

---

valid_encoded_rows/2

True when encoded training rows match the feature count induced by the encoders and carry numeric targets.

Compilation flags:
`static`

Template:
`valid_encoded_rows(Encoders,Rows)`
Mode and number of proofs:
`valid_encoded_rows(+list(compound),+list)` - `zero_or_one`

---

encoded_feature_count/2

Counts the number of numeric features induced by a list of continuous and categorical encoders, including missing-value indicator features.

Compilation flags:

```
static
```

Template:

```
encoded_feature_count(Encoders,FeatureCount)
```

Mode and number of proofs:

```
encoded_feature_count(+list(compound),-integer) - one
```

---

continuous_stats/5

Computes the mean and scaling factor used to encode a continuous attribute from the training examples according to the effective feature scaling option.

Compilation flags:

```
static
```

Template:

```
continuous_stats(Attribute,Examples,Options,Mean,Scale)
```

Mode and number of proofs:

```
continuous_stats(+atom,+list(compound),+list(compound),-float,-positive_float) -
one
```

---

examples_to_rows/3

Encodes dataset examples as numeric feature-vector and target pairs using a list of encoders.

Compilation flags:

```
static
```

Template:

```
examples_to_rows(Examples,Encoders,Rows)
```

Mode and number of proofs:

```
examples_to_rows(+list(compound),+list(compound),-list(pair)) - one
```

---

encode_instance/3

Validates and encodes an attribute-value list as a numeric feature vector using a list of continuous and categorical encoders.

Compilation flags:

```
static
```

Template:

```
encode_instance(Encoders,AttributeValues,Features)
```

Mode and number of proofs:

```
encode_instance(+list(compound),+list(pair),-list(float)) - one
```

---

fit_linear_model/7

Builds linear-model encoders from the training dataset, encodes the examples, fits an ordinary least-squares bias plus weight vector using the linear_algebra least-squares solver, and returns solver diagnostics terms.

Compilation flags:

```
static
```

Template:

```
fit_linear_model(Dataset,Options,Encoders,TrainingExampleCount,Bias,Weights,
TrainingDiagnostics)
```

Mode and number of proofs:

```
fit_linear_model(+object_identifier,+list(compound),-list(compound),-integer,
-float,-list(float),-list(compound)) - one
```

---

fit_ridge_model/7

Builds linear-model encoders from the training dataset, encodes the examples, fits a bias plus weight vector by solving the ridge normal equations with partial pivoting, and returns ridge-specific diagnostics terms.

Compilation flags:

```
static
```

Template:

```
fit_ridge_model(Dataset,Options,Encoders,TrainingExampleCount,Bias,Weights,
TrainingDiagnostics)
```

Mode and number of proofs:

```
fit_ridge_model(+object_identifier,+list(compound),-list(compound),-integer,
-float,-list(float),-list(compound)) - one
```

---

valid_feature_labels/1

True when a list of regression-tree feature labels only contains valid `feature/2` terms.

Compilation flags:
`static`

Template:
`valid_feature_labels(FeatureLabels)`
Mode and number of proofs:
`valid_feature_labels(+list(compound))` - `zero_or_one`

---

valid_regression_tree/2

True when a regression tree only contains valid `leaf/1` and `node/5` terms using feature indexes within bounds.

Compilation flags:
`static`

Template:
`valid_regression_tree(Tree,FeatureCount)`
Mode and number of proofs:
`valid_regression_tree(+compound,+positive_integer)` - `zero_or_one`

Private predicates

check_attribute_declarations/1

Checks that dataset attribute declarations use distinct attribute names and valid value domains.

Compilation flags:
`static`

Template:
`check_attribute_declarations(Attributes)`
Mode and number of proofs:
`check_attribute_declarations(+list(pair))` - `one_or_error`

Exceptions:
An attribute is declared more than once or uses an invalid value domain:
`domain_error(attribute_declarations,Attribute)`

---

known_attribute_values/3

Collects the known numeric values for a continuous attribute from the training examples, skipping omitted and variable values.

Compilation flags:
```
static
```

Template:
```
known_attribute_values(Examples,Attribute,Values)
```
Mode and number of proofs:
```
known_attribute_values(+list(compound),+atom,-list(number)) - one
```

---

check_attribute_bindings/2

Checks that an attribute-value list contains only declared attributes and does not repeat any declared attribute; omitted declared attributes are allowed and treated as missing values.

Compilation flags:
```
static
```

Template:
```
check_attribute_bindings(AttributeNames,AttributeValues)
```
Mode and number of proofs:
```
check_attribute_bindings(+list(atom),+list(pair)) - one
```

---

check_declared_attribute_bindings/2

Checks that no declared attribute appears more than once in an attribute-value list.

Compilation flags:
```
static
```

Template:
```
check_declared_attribute_bindings(AttributeNames,AttributeValues)
```
Mode and number of proofs:
```
check_declared_attribute_bindings(+list(atom),+list(pair)) - one
```

---

check_undeclared_attribute_bindings/2

Checks that every attribute occurring in an attribute-value list is declared.

Compilation flags:
```
static
```

Template:
```
check_undeclared_attribute_bindings(AttributeValues,AttributeNames)
```
Mode and number of proofs:
```
check_undeclared_attribute_bindings(+list(pair),+list(atom)) - one
```

---

attribute_occurrences/4

Counts the number of times an attribute occurs in an attribute-value list.

Compilation flags:
```
static
```

Template:
```
attribute_occurrences(AttributeValues,Attribute,Count0,Count)
```
Mode and number of proofs:
```
attribute_occurrences(+list(pair),+atom,+integer,-integer) - one
```

---

declared_attribute_names/2

Collects the declared attribute names from dataset attribute declarations.

Compilation flags:
```
static
```

Template:
```
declared_attribute_names(Attributes,AttributeNames)
```
Mode and number of proofs:
```
declared_attribute_names(+list(pair),-list(atom)) - one
```

---

encoder_attribute_names/2

Collects the declared attribute names from a list of regression encoders.

Compilation flags:
```
static
```

Template:
```
encoder_attribute_names(Encoders,AttributeNames)
```
Mode and number of proofs:
```
encoder_attribute_names(+list(compound),-list(atom)) - one
```

---

encode_instance_checked/3

Encodes an already validated attribute-value list as a numeric feature vector using a list of continuous and categorical encoders.

Compilation flags:
```
static
```

Template:
```
encode_instance_checked(Encoders,AttributeValues,Features)
```
Mode and number of proofs:
```
encode_instance_checked(+list(compound),+list(pair),-list(float)) - one
```

---

valid_linear_model_convergence/1

True when a linear-model optimization stop reason is recognized.

Compilation flags:
```
static
```

Template:
```
valid_linear_model_convergence(Convergence)
```
Mode and number of proofs:
```
valid_linear_model_convergence(+atom) - zero_or_one
```

---

normalize_continuous/4

Normalizes a continuous value using a stored mean and scale.

Compilation flags:
```
static
```

Template:
```
normalize_continuous(Value,Mean,Scale,Feature)
```
Mode and number of proofs:
```
normalize_continuous(+number,+float,+positive_float,-float) - one
```

---

check_categorical_value/3

Validates that a categorical value is declared for an attribute.

Compilation flags:
```
static
```

Template:
```
check_categorical_value(Attribute,Values,Value)
```
Mode and number of proofs:
```
check_categorical_value(+atom,+list,+nonvar) - one
```

---

one_hot_encode/4

Encodes a declared categorical value using reference-level dummy coding plus a trailing missing-value indicator feature set to zero.

Compilation flags:
```
static
```

Template:
```
one_hot_encode(Values,Value,Encoded,Tail)
```
Mode and number of proofs:
```
one_hot_encode(+list,+nonvar,-list(float),+list(float)) - one
```

---

missing_one_hot_encode/3

Encodes a missing categorical value as reference-level dummy zeroes plus a trailing missing-value indicator feature set to one.

Compilation flags:

```
static
```

Template:

```
missing_one_hot_encode(Values,Encoded,Tail)
```

Mode and number of proofs:

```
missing_one_hot_encode(+list,-list(float),+list(float)) - one
```

---

zero_vector_from_values/3

Creates a zero vector with one element per supplied categorical value slot.

Compilation flags:

```
static
```

Template:

```
zero_vector_from_values(Values,Zeroes,Tail)
```

Mode and number of proofs:

```
zero_vector_from_values(+list,-list(float),+list(float)) - one
```

---

build_linear_encoders/4

Builds the encoder list used by linear models from dataset attribute declarations, training examples, and the effective feature scaling option.

Compilation flags:

```
static
```

Template:

```
build_linear_encoders(Attributes,Examples,Options,Encoders)
```

Mode and number of proofs:

```
build_linear_encoders(+list(pair),+list(compound),+list(compound),
-list(compound)) - one
```

---

train_linear_model/6

Fits an ordinary least-squares linear model bias and weight vector from encoded rows by delegating the solve and rank estimation to the linear_algebra library and returns solver diagnostics terms.

Compilation flags:

```
static
```

Template:

```
train_linear_model(Rows,FeatureCount,Options,Bias,Weights,TrainingDiagnostics)
```

Mode and number of proofs:

```
train_linear_model(+list(pair),+integer,+list(compound),-float,-list(float),
-list(compound)) - one
```

---

rows_to_design_matrix/3

Transforms encoded training rows into a row-oriented design matrix with a leading intercept column and a target vector.

Compilation flags:

```
static
```

Template:

```
rows_to_design_matrix(Rows,DesignMatrix,Targets)
```

Mode and number of proofs:

```
rows_to_design_matrix(+list(pair),-list(list(float)),-list(float)) - one
```

---

residual_sum_of_squares/4

Computes the residual sum of squares for encoded training rows and a learned intercept plus weight vector.

Compilation flags:

```
static
```

Template:

```
residual_sum_of_squares(Rows,Bias,Weights,ResidualSumOfSquares)
```

Mode and number of proofs:

```
residual_sum_of_squares(+list(pair),+float,+list(float),-float) - one
```

---

train_ridge_model/6

Fits a ridge model bias and weight vector from encoded rows by solving a direct weighted linear system and returns diagnostics terms describing the solve.

Compilation flags:

```
static
```

Template:

```
train_ridge_model(Rows,FeatureCount,Options,Bias,Weights,TrainingDiagnostics)
```

Mode and number of proofs:

```
train_ridge_model(+list(pair),+integer,+list(compound),-float,-list(float),
-list(compound)) - one
```

---

ridge_feature_statistics/3

Computes per-feature ridge penalty weights and active-feature flags from encoded training rows, dropping zero-variance columns from the direct solve.

Compilation flags:

```
static
```

Template:

```
ridge_feature_statistics(Rows,PenaltyWeights,ActiveFlags)
```

Mode and number of proofs:

```
ridge_feature_statistics(+list(pair),-list(float),-list(atom)) - one
```

---

accumulate_feature_statistics/5

Accumulates per-feature sums and squared sums over encoded training rows.

Compilation flags:

```
static
```

Template:

```
accumulate_feature_statistics(Rows,Sums0,SumSquares0,Sums,SumSquares)
```

Mode and number of proofs:

```
accumulate_feature_statistics(+list(pair),+list(float),+list(float),-list(float),
-list(float)) - one
```

---

add_squared_vector/3

Adds the element-wise squares of a feature vector to an accumulated vector.

Compilation flags:

```
static
```

Template:

```
add_squared_vector(Features,AccumulatedSquares0,AccumulatedSquares)
```

Mode and number of proofs:

```
add_squared_vector(+list(float),+list(float),-list(float)) - one
```

---

feature_penalty_profiles/5

Builds ridge penalty weights and active-feature flags from per-feature sums and squared sums.

Compilation flags:

```
static
```

Template:

```
feature_penalty_profiles(Sums,SumSquares,Count,PenaltyWeights,ActiveFlags)
```

Mode and number of proofs:

```
feature_penalty_profiles(+list(float),+list(float),+integer,-list(float),
-list(atom)) - one
```

---

active_feature_count/2

Counts the number of active encoded features retained for the direct ridge solve.

Compilation flags:

```
static
```

Template:

```
active_feature_count(ActiveFlags,Count)
```

Mode and number of proofs:

```
active_feature_count(+list(atom),-integer) - one
```

---

compress_rows/3

Filters encoded training rows down to the active feature subset selected for the direct ridge solve.

Compilation flags:
```
static
```

Template:
```
compress_rows(Rows,ActiveFlags,CompressedRows)
```
Mode and number of proofs:
```
compress_rows(+list(pair),+list(atom),-list(pair)) - one
```

---

compress_features/3

Filters a feature vector down to the active feature subset selected for the direct ridge solve.

Compilation flags:
```
static
```

Template:
```
compress_features(Features,ActiveFlags,CompressedFeatures)
```
Mode and number of proofs:
```
compress_features(+list(float),+list(atom),-list(float)) - one
```

---

compress_vector/3

Filters a numeric vector down to the active positions selected for the direct ridge solve.

Compilation flags:
```
static
```

Template:
```
compress_vector(Vector,ActiveFlags,CompressedVector)
```
Mode and number of proofs:
```
compress_vector(+list(float),+list(atom),-list(float)) - one
```

---

build_ridge_system/5

Builds the regularized linear system for the intercept plus active ridge coefficients.

Compilation flags:

```
static
```

Template:

```
build_ridge_system(Rows,PenaltyWeights,Regularization,Matrix,Vector)
```

Mode and number of proofs:

```
build_ridge_system(+list(pair),+list(float),+float,-list(list(float)),
-list(float)) - one
```

---

accumulate_ridge_system/5

Accumulates the unregularized normal-equation matrix and target vector for a set of active encoded rows.

Compilation flags:

```
static
```

Template:

```
accumulate_ridge_system(Rows,Matrix0,Vector0,Matrix,Vector)
```

Mode and number of proofs:

```
accumulate_ridge_system(+list(pair),+list(list(float)),+list(float),
-list(list(float)),-list(float)) - one
```

---

add_outer_product/4

Adds the outer product of two vectors to an accumulated matrix.

Compilation flags:

```
static
```

Template:

```
add_outer_product(Vector1,Vector2,Matrix0,Matrix)
```

Mode and number of proofs:

```
add_outer_product(+list(float),+list(float),+list(list(float)),
-list(list(float))) - one
```

---

regularize_ridge_matrix/4

Adds the ridge penalty weights to the feature-feature diagonal block of a linear system matrix.

Compilation flags:
`static`

Template:
`regularize_ridge_matrix(Matrix0,PenaltyWeights,Regularization,Matrix)`
Mode and number of proofs:
`regularize_ridge_matrix(+list(list(float)),+list(float),+float,-list(list(float)))` - `one`

---

solve_linear_system/4

Solves a square linear system using partial pivoting Gaussian elimination and returns the solver name.

Compilation flags:
`static`

Template:
`solve_linear_system(Matrix,Vector,Solution,Solver)`
Mode and number of proofs:
`solve_linear_system(+list(list(float)),+list(float),-list(float),-atom)` - `one_or_error`

Exceptions:
The linear system is singular because a pivot coefficient becomes numerically zero:
`evaluation_error(zero_divisor)`

---

augment_rows/3

Pairs each matrix row with its corresponding right-hand-side value for elimination.

Compilation flags:
`static`

Template:
`augment_rows(Matrix,Vector,Rows)`
Mode and number of proofs:
`augment_rows(+list(list(float)),+list(float),-list(compound))` - `one`

---

triangularize/2

Transforms an augmented matrix into upper-triangular form using partial pivoting.

Compilation flags:
```
static
```

Template:
```
triangularize(Rows0,UpperRows)
```
Mode and number of proofs:
```
triangularize(+list(compound),-list(compound)) - one_or_error
```

Exceptions:
A pivot coefficient is numerically zero during elimination:
```
evaluation_error(zero_divisor)
```

---

select_pivot_row/3

Selects the pivot row with the largest leading magnitude and returns the remaining rows.

Compilation flags:
```
static
```

Template:
```
select_pivot_row(Rows0,PivotRow,RemainingRows)
```
Mode and number of proofs:
```
select_pivot_row(+list(compound),-compound,-list(compound)) - one
```

---

leading_magnitude/2

Returns the absolute leading coefficient magnitude of an augmented row.

Compilation flags:
```
static
```

Template:
```
leading_magnitude(Row,Magnitude)
```
Mode and number of proofs:
```
leading_magnitude(+compound,-float) - one
```

---

ensure_non_zero/1

Checks that a pivot coefficient is numerically non-zero.

Compilation flags:
`static`

Template:
`ensure_non_zero(Value)`
Mode and number of proofs:
`ensure_non_zero(+float)` - `one_or_error`

Exceptions:
A pivot coefficient is numerically zero:
`evaluation_error(zero_divisor)`

---

eliminate_rows/5

Eliminates the leading coefficient from remaining augmented rows using a pivot row.

Compilation flags:
`static`

Template:
`eliminate_rows(Pivot,PivotTail,PivotValue,Rows0,Rows)`
Mode and number of proofs:
`eliminate_rows(+float,+list(float),+float,+list(compound),-list(compound))` - `one`

---

scaled_row_difference/4

Subtracts a scaled pivot tail from another row tail.

Compilation flags:
`static`

Template:
`scaled_row_difference(PivotTail,RowTail,Factor,Difference)`
Mode and number of proofs:
`scaled_row_difference(+list(float),+list(float),+float,-list(float))` - `one`

---

back_substitution/2

Performs back-substitution on an upper-triangular augmented matrix.

Compilation flags:
```
static
```

Template:
```
back_substitution(UpperRows,Solution)
```
Mode and number of proofs:
```
back_substitution(+list(compound),-list(float)) - one_or_error
```

Exceptions:

A diagonal coefficient is numerically zero during back-substitution:
```
evaluation_error(zero_divisor)
```

---

maximum_linear_system_residual/4

Computes the maximum absolute residual of a solved linear system.

Compilation flags:
```
static
```

Template:
```
maximum_linear_system_residual(Matrix,Vector,Solution,MaximumResidual)
```
Mode and number of proofs:
```
maximum_linear_system_residual(+list(list(float)),+list(float),+list(float),
-float) - one
```

---

expand_weights/3

Expands active ridge coefficients back to the full encoded feature vector, inserting zeroes for dropped zero-variance features.

Compilation flags:
```
static
```

Template:
```
expand_weights(ActiveFlags,ActiveWeights,Weights)
```
Mode and number of proofs:
```
expand_weights(+list(atom),+list(float),-list(float)) - one
```

Operators

(none)
protocol

1.279.3 regressor_protocol

Protocol for machine learning regressors.

Availability:
`logtalk_load(regression_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-01

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/3*
  - *learn/2*
  - *predict/3*
  - *check_regressor/1*
  - *valid_regressor/1*
  - *diagnostics/2*
  - *diagnostic/2*
  - *regressor_options/2*
  - *export_to_clauses/4*
  - *export_to_file/4*
  - *print_regressor/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/3

Learns a regressor from the given dataset object using the specified options.

Compilation flags:
`static`

Template:
`learn(Dataset,Regressor,Options)`
Mode and number of proofs:
`learn(+object_identifier,-compound,+list(compound))` - `one`

---

learn/2

Learns a regressor from the given dataset object using default options.

Compilation flags:
`static`

Template:
`learn(Dataset,Regressor)`
Mode and number of proofs:
`learn(+object_identifier,-compound)` - `one`

---

predict/3

Predicts the numeric target value for a new instance using the learned regressor. The instance is a list of `Attribute-Value` pairs.

Compilation flags:
`static`

Template:
`predict(Regressor,Instance,Target)`
Mode and number of proofs:
`predict(+compound,+list,-number)` - `one`

---

check_regressor/1

Checks that a learned regressor term is structurally valid for the receiving implementation. Throws an exception when the term is not a valid regressor representation.

Compilation flags:
`static`

Template:
`check_regressor(Regressor)`
Mode and number of proofs:
`check_regressor(@compound)` - `one_or_error`

Exceptions:
`Regressor` is a variable:
`instantiation_error`
`Regressor` is neither a variable nor a valid regressor:
`domain_error(regressor,Regressor)`

---

valid_regressor/1

True when a learned regressor term is structurally valid for the receiving implementation. Succeeds iff `check_regressor/1` succeeds without throwing an exception.

Compilation flags:
`static`

Template:
`valid_regressor(Regressor)`
Mode and number of proofs:
`valid_regressor(@compound)` - `zero_or_one`

---

diagnostics/2

Returns diagnostics metadata for a learned regressor.

Compilation flags:
`static`

Template:
`diagnostics(Regressor,Diagnostics)`
Mode and number of proofs:
`diagnostics(+compound,-list(compound))` - `one`

---

diagnostic/2

Enumerates individual diagnostics metadata terms for a learned regressor.

Compilation flags:
`static`

Template:
`diagnostic(Regressor,Diagnostic)`
Mode and number of proofs:
`diagnostic(+compound,?compound)` - `zero_or_more`

---

regressor_options/2

Returns the effective options stored in a learned regressor diagnostics metadata.

Compilation flags:
`static`

Template:
`regressor_options(Regressor,Options)`
Mode and number of proofs:
`regressor_options(+compound,-list(compound))` - `one`

---

export_to_clauses/4

Converts a regressor into a list of predicate clauses. `Functor` is the functor for the generated predicate clauses. When exporting a serialized regressor term, a noun such as `regressor` or `model` is usually clearer than a verb such as `predict`.

Compilation flags:
`static`

Template:
`export_to_clauses(Dataset,Regressor,Functor,Clauses)`
Mode and number of proofs:
`export_to_clauses(+object_identifier,+compound,+callable,-list(clause))` - `one`

---

export_to_file/4

Exports a regressor to a file. `Functor` is the functor for the generated predicate clauses. When exporting a serialized regressor term, a noun such as `regressor` or `model` is usually clearer than a verb such as `predict`.

Compilation flags:
`static`

Template:
`export_to_file(Dataset,Regressor,Functor,File)`
Mode and number of proofs:
`export_to_file(+object_identifier,+compound,+callable,+atom)` - `one`

---

print_regressor/1

Prints a regressor to the current output stream in a human-readable format.

Compilation flags:
`static`

Template:
`print_regressor(Regressor)`
Mode and number of proofs:
`print_regressor(+compound)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

linear_regression, *regression_dataset_protocol*

1.280 regression_tree

object

1.280.1 `regression_tree`

Regression tree regressor supporting continuous and mixed-feature datasets using recursive variance-reduction splits.

Availability:
`logtalk_load(regression_tree(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *regressor_common*
Uses:
fast_random(Algorithm)
format
list
population
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

linear_regression, *knn_regression*, *random_forest_regression*, *gradient_boosting_regression*

1.281 regularized_bradley_terry_ranker

object

1.281.1 `regularized_bradley_terry_ranker`

Regularized Bradley-Terry MAP ranker. Learns one positive strength parameter per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol using a Bradley-Terry likelihood with an explicit independent Gamma prior, and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:

```
logtalk_load(regularized_bradley_terry_ranker(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public ranking_dataset_common
public pairwise_strength_ranker_common
public score_ranker_common
```

Uses:

avltree
format
list

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *bradley_terry_ranker*

1.282 relief_feature_selector

object

1.282.1 `relief_feature_selector`

Binary Relief feature selection with one nearest hit and miss.

Availability:
`logtalk_load(relief_feature_selector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *relief_feature_selector_common*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_selector/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *feature_scores/2* *learn/2* *learn/3* *option/2* *option/3* *print_selector/1* *selected_features/2* *selector_options/2* *valid_option/1* *valid_options/1* *valid_selector/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.283 relieff_feature_selector

object

1.283.1 `relieff_feature_selector`

Multiclass ReliefF feature selection with class-prior miss weighting.

Availability:
`logtalk_load(relieff_feature_selector(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *relief_feature_selector_common*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_selector/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *feature_scores/2* *learn/2* *learn/3* *option/2* *option/3* *print_selector/1* *selected_features/2* *selector_options/2* *valid_option/1* *valid_options/1* *valid_selector/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.284 rest

category

1.284.1 `rest`

REST authoring layer built on top of the HTTP router category.

Availability:
`logtalk_load(rest(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-09-03

Compilation flags:
`static`

Extends:
`public` *http_router*
Uses:
http_core
list
open_api

Remarks:
(none)

Inherited public predicates:
api_info/1 *handle/2* *operations/1* *schema/2* *security/1* *security_scheme/2* *servers/1*

- *Public predicates*
  - *path_parameter/3*

      - *query_parameter/3*
      - *request_header/3*
      - *request_body/2*
      - *json_body/2*
      - *json_object_body/2*
      - *json_array_body/2*
      - *form_body/2*
      - *text_body/2*
      - *binary_body/2*
      - *json_response/4*
      - *json_response/5*
      - *created_response/4*
      - *no_content_response/2*
      - *problem_response/6*
    - *Protected predicates*
      - *endpoint/5*
      - *open_api_validate_request/1*
      - *open_api_request_validation_error_response/4*
      - *open_api_validate_response/1*
      - *open_api_response_validation_error_response/5*
      - *rest_action_failure_response/3*
      - *rest_action_error_response/4*
      - *dispatch_rest_endpoint/2*
    - *Private predicates*
    - *Operators*

Public predicates

path_parameter/3

Returns a named path parameter extracted by the router from the current request using deterministic lookup.

Compilation flags:

```
static
```

Template:

```
path_parameter(Request,Name,Value)
```

Mode and number of proofs:

```
path_parameter(+compound,?atom,?term) - zero_or_one
```

query_parameter/3

Returns the first named query parameter from the current request when query pairs are available.

Compilation flags:
`static`

Template:
`query_parameter(Request,Name,Value)`
Mode and number of proofs:
`query_parameter(+compound,?atom,?term)` - `zero_or_one`

---

request_header/3

Returns a normalized request header value.

Compilation flags:
`static`

Template:
`request_header(Request,Name,Value)`
Mode and number of proofs:
`request_header(+compound,?atom,?term)` - `zero_or_more`

---

request_body/2

Returns the normalized request body term.

Compilation flags:
`static`

Template:
`request_body(Request,Body)`
Mode and number of proofs:
`request_body(+compound,-compound)` - `zero_or_one`

---

json_body/2

Returns the decoded JSON payload from a normalized request body and reports a client error when the request body is missing or not decoded as JSON.

Compilation flags:

```
static
```

Template:

```
json_body(Request,JSON)
```

Mode and number of proofs:

```
json_body(+compound,-term) - one_or_error
```

Exceptions:

The normalized request body is missing or is not decoded as JSON:

```
problem(400,urn:logtalk:invalid-request-body,Bad Request,Expected JSON
request body.)
```

---

json_object_body/2

Returns the decoded JSON payload when it is a JSON object term and reports a client error when the request body is missing, not decoded as JSON, or not an object.

Compilation flags:

```
static
```

Template:

```
json_object_body(Request,JSONObject)
```

Mode and number of proofs:

```
json_object_body(+compound,-term) - one_or_error
```

Exceptions:

The normalized request body is missing or is not decoded as JSON:

```
problem(400,urn:logtalk:invalid-request-body,Bad Request,Expected JSON
request body.)
```

The decoded JSON payload is not a JSON object term:

```
problem(400,urn:logtalk:invalid-request-body,Bad Request,Expected JSON object
request body.)
```

---

json_array_body/2

Returns the decoded JSON payload when it is a JSON array term and reports a client error when the request body is missing, not decoded as JSON, or not an array.

Compilation flags:

```
static
```

Template:

```
json_array_body(Request,JSONArray)
```

Mode and number of proofs:

```
json_array_body(+compound,-list) - one_or_error
```

Exceptions:

The normalized request body is missing or is not decoded as JSON:

```
problem(400,urn:logtalk:invalid-request-body,Bad Request,Expected JSON
request body.)
```

The decoded JSON payload is not a JSON array term:

```
problem(400,urn:logtalk:invalid-request-body,Bad Request,Expected JSON array
request body.)
```

---

form_body/2

Returns the decoded form payload from a normalized request body and reports a client error when the request body is missing or not decoded as form data.

Compilation flags:

```
static
```

Template:

```
form_body(Request,Pairs)
```

Mode and number of proofs:

```
form_body(+compound,-list(compound)) - one_or_error
```

Exceptions:

The normalized request body is missing or is not decoded as form data:

```
problem(400,urn:logtalk:invalid-request-body,Bad Request,Expected form
request body.)
```

---

text_body/2

Returns the decoded text payload from a normalized request body and reports a client error when the request body is missing or not decoded as text.

Compilation flags:

```
static
```

Template:

```
text_body(Request,Text)
```

Mode and number of proofs:

```
text_body(+compound,-atom) - one_or_error
```

Exceptions:

The normalized request body is missing or is not decoded as text:

```
problem(400,urn:logtalk:invalid-request-body,Bad Request,Expected text
request body.)
```

---

binary_body/2

Returns the decoded binary payload from a normalized request body and reports a client error when the request body is missing or not decoded as binary.

Compilation flags:

```
static
```

Template:

```
binary_body(Request,Bytes)
```

Mode and number of proofs:

```
binary_body(+compound,-list(integer)) - one_or_error
```

Exceptions:

The normalized request body is missing or is not decoded as binary:

```
problem(400,urn:logtalk:invalid-request-body,Bad Request,Expected binary
request body.)
```

---

json_response/4

Builds a JSON response for the given request and status descriptor, honoring any negotiated response media type annotation.

Compilation flags:

```
static
```

Template:

```
json_response(Request,Status,JSON,Response)
```

Mode and number of proofs:

`json_response(+compound,++term,++term,-compound)` - `one_or_error`

Exceptions:

`Status` is a variable or a `status(Code, Reason)` term with a variable argument:

`instantiation_error`

`Status` is an integer without a built-in reason phrase supported by the rest library:

`domain_error(http_status_code,Status)`

`Status` is neither an integer nor a valid normalized HTTP status term:

`domain_error(http_status,Status)`

`json_response/5`

Builds a JSON response for the given request and status descriptor, honoring any negotiated response media type annotation and adding the provided normalized headers.

Compilation flags:

`static`

Template:

`json_response(Request,Status,Headers,JSON,Response)`

Mode and number of proofs:

`json_response(+compound,++term,+list(compound),++term,-compound)` - `one_or_error`

Exceptions:

`Status` is a variable or a `status(Code, Reason)` term with a variable argument:

`instantiation_error`

`Status` is an integer without a built-in reason phrase supported by the rest library:

`domain_error(http_status_code,Status)`

`Status` is neither an integer nor a valid normalized HTTP status term:

`domain_error(http_status,Status)`

`Headers` is not a valid normalized header list:

`domain_error(http_headers,Headers)`

`created_response/4`

Builds a `201 Created` JSON response with a `Location` header, honoring any negotiated response media type annotation.

Compilation flags:

`static`

Template:

`created_response(Request,Location,JSON,Response)`

Mode and number of proofs:

`created_response(+compound,++term,++term,-compound)` - `one_or_error`

Exceptions:

`Location` is not a valid normalized header value:

`domain_error(http_header_value,Location)`

`no_content_response/2`

Builds a `204 No Content` response.

Compilation flags:

`static`

Template:

`no_content_response(Request,Response)`

Mode and number of proofs:

`no_content_response(+compound,-compound)` - `one_or_error`

Exceptions:

`Request` is not a valid normalized HTTP request term:

`domain_error(http_request,Request)`

`problem_response/6`

Builds an `application/problem+json` response with the standard `type`, `title`, `detail`, and numeric `status` fields.

Compilation flags:

`static`

Template:

`problem_response(Request,Status,Type,Title,Detail,Response)`

Mode and number of proofs:

`problem_response(+compound,++term,++atom,++atom,++atom,-compound)` - `one_or_error`

Exceptions:

`Status` is a variable or a `status(Code, Reason)` term with a variable argument:

`instantiation_error`

`Status` is an integer without a built-in reason phrase supported by the rest library:

`domain_error(http_status_code,Status)`

`Status` is neither an integer nor a valid normalized HTTP status term:

`domain_error(http_status,Status)`

Protected predicates

endpoint/5

Hook predicate that enumerates REST endpoint descriptors. `Id` is the endpoint identifier and must be unique within the importing object; when OpenAPI derivation is used it also becomes the operation identifier. `Method` is the lowercase HTTP method, `Path` the path template, `Action` the name of a declared local predicate with arity 2, and `Options` a list of route metadata terms plus the optional `produces(MediaTypes)` descriptor.

Compilation flags:

```
static
```

Template:

```
endpoint(Id,Method,Path,Action,Options)
```

Mode and number of proofs:

```
endpoint(?atom,?atom,?atom,?atom,?list(compound)) - zero_or_more
```

---

open_api_validate_request/1

Optional hook predicate that enables automatic OpenAPI request contract validation for the given endpoint identifier before its action predicate is called.

Compilation flags:

```
static
```

Template:

```
open_api_validate_request(Id)
```

Mode and number of proofs:

```
open_api_validate_request(?atom) - zero_or_more
```

---

open_api_request_validation_error_response/4

Optional hook predicate that customizes the response returned when OpenAPI request contract validation fails. The default implementation preserves the distinction between generic `400` request failures, `415 Unsupported Media Type`, and `422 Unprocessable Content` for invalid request bodies. The arguments are the current request, the endpoint identifier, the validation errors, and the response to send.

Compilation flags:

```
static
```

Template:

```
open_api_request_validation_error_response(Request,Id,Errors,Response)
```

Mode and number of proofs:

```
open_api_request_validation_error_response(+compound,+atom,+list,-compound) -
zero_or_one
```

---

open_api_validate_response/1

Optional hook predicate that enables automatic OpenAPI response contract validation for the given endpoint identifier after its action result is normalized into an HTTP response.

Compilation flags:

```
static
```

Template:

```
open_api_validate_response(Id)
```

Mode and number of proofs:

```
open_api_validate_response(?atom) - zero_or_more
```

---

open_api_response_validation_error_response/5

Optional hook predicate that customizes the response returned when OpenAPI response contract validation fails. The arguments are the current request, the endpoint identifier, the candidate response, the validation errors, and the response to send.

Compilation flags:

```
static
```

Template:

```
open_api_response_validation_error_response(Request,Id,Response0,Errors,Response)
```

Mode and number of proofs:

```
open_api_response_validation_error_response(+compound,+atom,+compound,+list,
-compound) - zero_or_one
```

---

rest_action_failure_response/3

Optional hook predicate that customizes the response returned when the matched endpoint action fails without throwing an exception. The arguments are the current request, the endpoint identifier, and the response to send.

Compilation flags:

```
static
```

Template:

```
rest_action_failure_response(Request,Id,Response)
```

Mode and number of proofs:

```
rest_action_failure_response(+compound,+atom,-compound) - zero_or_one
```

---

rest_action_error_response/4

Optional hook predicate that customizes the response returned when the matched endpoint action throws an unsupported exception. The arguments are the current request, the endpoint identifier, the thrown term, and the response to send.

Compilation flags:

```
static
```

Template:

```
rest_action_error_response(Request,Id,Error,Response)
```

Mode and number of proofs:

```
rest_action_error_response(+compound,+atom,+term,-compound) - zero_or_one
```

---

dispatch_rest_endpoint/2

Runs the action associated with the matched endpoint and normalizes any supported returned or thrown result term into an HTTP response.

Compilation flags:

```
static
```

Template:

```
dispatch_rest_endpoint(Request,Response)
```

Mode and number of proofs:

```
dispatch_rest_endpoint(+compound,-compound) - one_or_error
```

Exceptions:

The routed endpoint identifier is missing or unknown:

```
existence_error(rest_endpoint_id,Id)
```

The routed endpoint identifier is duplicated:

```
domain_error(rest_endpoint_id,duplicate(Id))
```

The endpoint action produces an invalid REST result term:

```
domain_error(rest_result,Result)
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.285 ridge_regression

object

1.285.1 `ridge_regression`

Ridge regression regressor supporting continuous and mixed-feature datasets using a direct weighted linear-system solve with L2 regularization. Learns from a dataset object implementing the `regression_dataset_protocol` protocol and returns a regressor term that can be used for prediction and exported as predicate clauses.

Availability:
`logtalk_load(ridge_regression(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *regressor_common*
Uses:
format
list
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

linear_regression, *knn_regression*, *regression_tree*, *random_forest_regression*, *gradient_boosting_regression*

1.286 ripple_down_rules

object

1.286.1 `generalized_ripple_down_rules`

Generalized Ripple-Down Rules implementation using keyed Single-Classification Ripple-Down Rules and Multi-Classification Ripple-Down Rules submodels.

Availability:

```
logtalk_load(ripple_down_rules(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *ripple_down_rules_common_protocol*

Imports:

`public` *options*

Uses:

avltree
multi_classification_ripple_down_rules
single_classification_ripple_down_rules

Remarks:

(none)

Inherited public predicates:

as_list/2 check_model/1 check_option/1 check_options/1 classify/3 classify/4 default_option/1 default_options/1 new/1 new/2 option/2 option/3 revise/6 size/2 valid_model/1 valid_option/1 valid_options/1

- *Public predicates*
  - *put/4*
  - *get/3*
  - *remove/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

put/4

Returns a new Generalized Ripple-Down Rules model with the keyed Single-Classification Ripple-Down Rules or Multi-Classification Ripple-Down Rules submodel added or replaced.

Compilation flags:

```
static
```

Template:

```
put(Model,Key,Submodel,NewModel)
```

Mode and number of proofs:

```
put(+compound,+term,+compound,-compound) - one_or_error
```

Exceptions:

`Model` is a variable:

```
instantiation_error
```

`Model` is neither a variable nor a valid model:

```
domain_error(ripple_down_rules,Model)
```

`Key` is a variable:

```
instantiation_error
```

`Submodel` is a variable:

```
instantiation_error
```

`Submodel` is neither a variable nor a valid submodel:

```
domain_error(ripple_down_rules_submodel,Submodel)
```

get/3

Returns the submodel stored under a key.

Compilation flags:
`static`

Template:
`get(Model,Key,Submodel)`
Mode and number of proofs:
`get(+compound,+term,-compound)` - `zero_or_one_or_error`

Exceptions:
`Model` is a variable:
`instantiation_error`
`Model` is neither a variable nor a valid model:
`domain_error(ripple_down_rules,Model)`
`Key` is a variable:
`instantiation_error`

---

remove/3

Returns a new Generalized Ripple-Down Rules model with the keyed submodel removed.

Compilation flags:
`static`

Template:
`remove(Model,Key,NewModel)`
Mode and number of proofs:
`remove(+compound,+term,-compound)` - `zero_or_one_or_error`

Exceptions:
`Model` is a variable:
`instantiation_error`
`Model` is neither a variable nor a valid model:
`domain_error(ripple_down_rules,Model)`
`Key` is a variable:
`instantiation_error`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.286.2 `multi_classification_ripple_down_rules`

Multi-Classification Ripple-Down Rules implementation.

Availability:
`logtalk_load(ripple_down_rules(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-06

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ripple_down_rules_protocol*
Imports:
`public` *options*
Uses:
list

Remarks:
(none)

Inherited public predicates:
as_list/2 check_model/1 check_option/1 check_options/1 classify/3 classify/4 classify_with_context/5 default_option/1 default_options/1 new/1 new/2 option/2 option/3 revise/6 size/2 valid_model/1 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.286.3 `ripple_down_rules_common_protocol`

Common orotocol for Ripple-Down Rules models.

Availability:
`logtalk_load(ripple_down_rules(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *new/1*
  - *new/2*
  - *classify/3*
  - *classify/4*
  - *revise/6*
  - *as_list/2*
  - *size/2*
  - *check_model/1*

    - *valid_model/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new/1

Creates an empty Ripple-Down Rules model using default options.

Compilation flags:
`static`

Template:
`new(Model)`
Mode and number of proofs:
`new(-compound)` - `one`

---

new/2

Creates an empty Ripple-Down Rules model using the given options.

Compilation flags:
`static`

Template:
`new(Model,Options)`
Mode and number of proofs:
`new(-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

classify/3

Classifies a case using a Ripple-Down Rules model.

Compilation flags:
`static`

Template:
`classify(Model,Case,Conclusion)`
Mode and number of proofs:
`classify(+compound,@term,--term)` - `one`

---

classify/4

Classifies a case and returns an ordered inference trace.

Compilation flags:
`static`

Template:
`classify(Model,Case,Conclusion,Trace)`
Mode and number of proofs:
`classify(+compound,@term,--term,-list(compound))` - `one`

---

revise/6

Returns a new model after applying one correction. The `scrdr` implementation accepts the correction atom `replace`. The `mcrdr` implementation accepts the correction atoms `add`, `remove`, and `filter`. The `grdr` implementation accepts the correction atoms supported by the keyed submodel and requires the second argument to be a `Key-Case` pair. The condition closure is called with the case and current conclusions. The conclusion closure is called with the case and current conclusions plus an output argument.

Compilation flags:
`static`

Template:
`revise(Model,Case,Correction,Condition,Conclusion,NewRDR)`
Meta-predicate template:
`revise(*,*,*,2,3,*)`
Mode and number of proofs:
`revise(+compound,@term,+atom,+callable,+callable,-compound)` - `one_or_error`

Exceptions:
`Correction` is a variable:
`instantiation_error`
`Correction` is not supported by the receiving implementation or keyed submodel:

```
domain_error(ripple_down_rule_correction,Correction)
```

---

as_list/2

Returns stable preorder descriptors for all rules in a model.

Compilation flags:

```
static
```

Template:

```
as_list(Model,Rules)
```

Mode and number of proofs:

```
as_list(+compound,-list(compound)) - one
```

---

size/2

Returns the number of rules in a model.

Compilation flags:

```
static
```

Template:

```
size(Model,Size)
```

Mode and number of proofs:

```
size(+compound,-integer) - one
```

---

check_model/1

Checks that a term is a structurally valid model for the receiving implementation.

Compilation flags:

```
static
```

Template:

```
check_model(Model)
```

Mode and number of proofs:

```
check_model(@compound) - one_or_error
```

Exceptions:

`Model` is a variable:

```
instantiation_error
```

`Model` is neither a variable nor a valid model:

```
domain_error(ripple_down_rules_model,Model)
```

`valid_model/1`

True iff the argument is a structurally valid model for the receiving implementation.

Compilation flags:
`static`

Template:
`valid_model(Model)`
Mode and number of proofs:
`valid_model(@compound)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
protocol

1.286.4 `ripple_down_rules_protocol`

Protocol for Ripple-Down Rules models.

Availability:
`logtalk_load(ripple_down_rules(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-06

Compilation flags:
`static`

Extends:
`public` *ripple_down_rules_common_protocol*

Remarks:
(none)

Inherited public predicates:
as_list/2 *check_model/1* *classify/3* *classify/4* *new/1* *new/2* *revise/6* *size/2* *valid_model/1*

- *Public predicates*
  - *classify_with_context/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

classify_with_context/5

Classifies a case while making external keyed conclusions visible to rule closures.

Compilation flags:
```
static
```

Template:
```
classify_with_context(Model,Case,Context,Conclusion,Trace)
```
Mode and number of proofs:
```
classify_with_context(+compound,@term,+list(pair),--term,-list(compound)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.286.5 single_classification_ripple_down_rules

Single-Classification Ripple-Down Rules implementation.

Availability:
```
logtalk_load(ripple_down_rules(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-06

Compilation flags:
```
static, context_switching_calls
```

Implements:
`public` *ripple_down_rules_protocol*
Imports:
`public` *options*
Uses:
list

Remarks:
(none)

Inherited public predicates:
as_list/2 *check_model/1* *check_option/1* *check_options/1* *classify/3* *classify/4* *classify_with_context/5* *default_option/1* *default_options/1* *new/1* *new/2* *option/2* *option/3* *revise/6* *size/2* *valid_model/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.287 rrelieff_feature_selector

object

1.287.1 `rrelieff_feature_selector`

RReliefF regression feature selection using normalized conditional difference masses.

Availability:
`logtalk_load(rrelieff_feature_selector(loader))`

Author: Paulo Moura
Version: 1:0:0

Date: 2026-10-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *relief_feature_selector_common*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_selector/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *feature_scores/2* *learn/2* *learn/3* *option/2* *option/3* *print_selector/1* *selected_features/2* *selector_options/2* *valid_option/1* *valid_options/1* *valid_selector/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.288 s3

object

1.288.1 s3_client

Stateless S3-compatible client facade over the normalized HTTP client using by default the `http_process_transport` library.

Availability:
`logtalk_load(s3(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *s3_client(http_process_transport)*

Remarks:
(none)

Inherited public predicates:
check_option/1 check_options/1 copy_object/5 default_option/1 default_options/1 delete_object/4 get_object/5 head_bucket/3 head_object/4 list_buckets/2 list_objects_v2/4 option/2 option/3 presigned_get_object/4 presigned_post_object/4 presigned_put_object/4 put_object/6 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.288.2 `s3_client(HTTPTransport)`

Stateless S3-compatible client facade over the normalized HTTP client.

Availability:
`logtalk_load(s3(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *s3_protocol*
Imports:
`public` *options*
Uses:
http_client
http_core
list
mime_types
os
s3_endpoints
s3_headers
s3_signing
s3_xml
type
url(Representation)
user

Remarks:
(none)

Inherited public predicates:
check_option/1 check_options/1 copy_object/5 default_option/1 default_options/1 delete_object/4 get_object/5 head_bucket/3 head_object/4 list_buckets/2 list_objects_v2/4 option/2 option/3 presigned_get_object/4 presigned_post_object/4 presigned_put_object/4 put_object/6 valid_option/1 valid_options/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.288.3 `s3_endpoints`

Endpoint and canonical-URI helpers for S3-compatible services.

Availability:
`logtalk_load(s3(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
list
url(Representation)
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *request/6*
  - *canonical_uri/4*
  - *canonical_uri/5*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

request/6

Builds the request base URL plus signing endpoint context for an S3 operation.

Compilation flags:
`static`

Template:
`request(Operation,Bucket,Key,URL,Endpoint,Options)`

Mode and number of proofs:
`request(+atom,@atom,@atom,-atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Operation` is not a supported S3 operation:
`domain_error(s3_operation,Operation)`
`Bucket` is a variable:
`instantiation_error`
`Bucket` is neither a variable not a valid bucket:
`domain_error(s3_request,Bucket)`
`Key` is a variable:
`instantiation_error`
`Key` is neither a variable not a valid key:
`domain_error(s3_request,Key)`
`Options` contains an invalid endpoint or addressing style:
`domain_error(s3_endpoint,Endpoint)`

Remarks:

- Option `region/1`: AWS region to use when `endpoint/1` is absent. Defaults to `us-east-1`.
- Option `endpoint/1`: Optional custom S3-compatible base endpoint URL, which may include a path prefix.
- Option `addressing_style/1`: Endpoint addressing style, either `virtual_hosted` or `path`. Defaults to `virtual_hosted`.

---

canonical_uri/4

Returns the canonical request URI for an S3 operation using the selected endpoint style.

Compilation flags:
`static`

Template:
`canonical_uri(Operation,Bucket,Key,CanonicalURI)`

Mode and number of proofs:

`canonical_uri(+atom,@atom,@atom,-atom)` - `one_or_error`

Exceptions:

`Operation` is not a supported S3 operation:

`domain_error(s3_operation,Operation)`

`Bucket` is a variable:

`instantiation_error`

`Bucket` is neither a variable not a valid bucket:

`domain_error(s3_request,Bucket)`

`Key` is a variable:

`instantiation_error`

`Key` is neither a variable not a valid key:

`domain_error(s3_request,Key)`

`canonical_uri/5`

Returns the canonical request URI for an S3 operation using an explicit endpoint context.

Compilation flags:

`static`

Template:

`canonical_uri(Operation,Context,Bucket,Key,CanonicalURI)`

Mode and number of proofs:

`canonical_uri(+atom,+compound,@atom,@atom,-atom)` - `one_or_error`

Exceptions:

`Operation` is not a supported S3 operation:

`domain_error(s3_operation,Operation)`

`Context` is not a valid endpoint context:

`domain_error(s3_endpoint,Context)`

`Bucket` is a variable:

`instantiation_error`

`Bucket` or `Key` is invalid for the operation:

`domain_error(s3_request,Argument)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.288.4 `s3_headers`

Helpers for extracting S3 response metadata from normalized HTTP responses.

Availability:
`logtalk_load(s3(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:
`static, context_switching_calls`

Uses:
http_core

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *response_properties/2*
  - *header_value/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

response_properties/2

Collects status and response headers into a stable property list.

Compilation flags:
`static`

Template:
`response_properties(Response,Properties)`
Mode and number of proofs:
`response_properties(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Response` is not a normalized HTTP response term:
`domain_error(http_response,Response)`

---

header_value/3

Enumerates response header values by normalized header name.

Compilation flags:
`static`

Template:
`header_value(Response,Name,Value)`
Mode and number of proofs:
`header_value(+compound,+atom,-term)` - `zero_or_more`

Exceptions:
`Response` is not a normalized HTTP response term:
`domain_error(http_response,Response)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.288.5 `s3_protocol`

S3-compatible client facade predicates.

Availability:
`logtalk_load(s3(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *list_buckets/2*
  - *head_bucket/3*
  - *list_objects_v2/4*
  - *head_object/4*
  - *get_object/5*
  - *put_object/6*
  - *delete_object/4*
  - *copy_object/5*
  - *presigned_get_object/4*
  - *presigned_put_object/4*
  - *presigned_post_object/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

list_buckets/2

Lists accessible buckets.

Compilation flags:
`static`

Template:
`list_buckets(Buckets,Options)`

Mode and number of proofs:
`list_buckets(-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Options` contains an invalid S3 client option:
`domain_error(s3_client_option,Option)`
Credentials are missing or invalid:
`domain_error(s3_credentials,missing)`
The S3 service returned a non-success status or malformed XML body:
`domain_error(s3_http_status,Status)`

Remarks:

- Option `credentials/1`: Required access credentials as `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`.
- Option `session_token/1`: Optional session token for temporary credentials.
- Option `region/1`: AWS region to use when `endpoint/1` is absent. Defaults to `us-east-1`.
- Option `endpoint/1`: Optional custom S3-compatible endpoint URL.
- Option `addressing_style/1`: Endpoint addressing style, either `virtual_hosted` or `path`. Defaults to `virtual_hosted`.
- Option `headers/1`: Optional extra normalized HTTP request headers.
- Option `version/1`: HTTP protocol version as `http(Major, Minor)`. Defaults to `http(1,1)`.
- Option `properties/1`: Optional extra normalized HTTP request properties.
- Option `connection_options/1`: Optional transport-specific HTTP client options.
- Option `payload_hash_mode/1`: Payload signing mode, either `signed` or `unsigned`. Defaults to `signed`.
- Option `request_time/1`: Optional explicit local request time as `date_time(Year, Month, Day, Hours, Minutes, Seconds)`.
- Option `utc_offset_seconds/1`: UTC offset used to derive the signing time from the local clock. Defaults to `0`.
- Option `amz_date/1`: Optional explicit AWS SigV4 timestamp overriding any derived request time.

head_bucket/3

Returns response metadata for a bucket.

Compilation flags:

`static`

Template:

`head_bucket(Bucket,Metadata,Options)`

Mode and number of proofs:

`head_bucket(+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Bucket` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid S3 client option:

`domain_error(s3_client_option,Option)`

The S3 service returned a non-success status:

`domain_error(s3_http_status,Status)`

Remarks:

- Options: Supported options are the standard S3 request options: `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`, optional `session_token(Token)`, `region(Region)`, optional custom `endpoint(URL)`, `addressing_style(virtual_hosted|path)`, optional `headers(Headers)`, `version(http(Major, Minor))`, optional `properties(Properties)`, optional `connection_options(ConnectionOptions)`, `payload_hash_mode(signed|unsigned)`, optional `request_time(date_time(...))`, `utc_offset_seconds(OffsetSeconds)`, and optional explicit `amz_date(AmzDate)`.

list_objects_v2/4

Lists bucket objects using the ListObjectsV2 API.

Compilation flags:

`static`

Template:

`list_objects_v2(Bucket,Request,Listing,Options)`

Mode and number of proofs:

`list_objects_v2(+atom,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Bucket` is a variable:

`instantiation_error`

`Request` is a variable:

`instantiation_error`

An element `Option` of the request option list is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

An element `Option` of the request option list is neither a variable nor a compound term:

`type_error(compound,Option)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Request` is not a valid ListObjectsV2 request term or option list:

`domain_error(s3_list_objects_request,Request)`

An element `Option` of the request option list is a compound term but not a valid ListObjectsV2 request option:

`domain_error(s3_list_objects_request_option,Option)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid S3 client option:

`domain_error(s3_client_option,Option)`

The S3 service returned a non-success status or malformed XML body:

`domain_error(s3_http_status,Status)`

Remarks:

- Options: `Options` accepts the standard S3 request options: `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`, optional `session_token(Token)`, `region(Region)`, optional custom `endpoint(URL)`, `addressing_style(virtual_hosted|path)`, optional `headers(Headers)`, `version(http(Major, Minor))`, optional `properties(Properties)`, optional

`connection_options(ConnectionOptions)`, `payload_hash_mode(signed|unsigned)`, optional `request_time(date_time(...))`, `utc_offset_seconds(OffsetSeconds)`, and optional explicit `amz_date(AmzDate)`. The `Request` argument carries the ListObjectsV2-specific request options.

`head_object/4`

Returns response metadata for an object.

Compilation flags:

`static`

Template:

`head_object(Bucket,Key,Metadata,Options)`

Mode and number of proofs:

`head_object(+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Bucket` is a variable:

`instantiation_error`

`Key` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid S3 client option:

`domain_error(s3_client_option,Option)`

The S3 service returned a non-success status:

`domain_error(s3_http_status,Status)`

Remarks:

- Options: Supported options are the standard S3 request options: `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`, optional `session_token(Token)`, `region(Region)`, optional custom `endpoint(URL)`, `addressing_style(virtual_hosted|path)`, optional `headers(Headers)`, `version(http(Major, Minor))`, optional `properties(Properties)`, optional `connection_options(ConnectionOptions)`, `payload_hash_mode(signed|unsigned)`, optional

`request_time(date_time(...))`, `utc_offset_seconds(OffsetSeconds)`, and optional explicit `amz_date(AmzDate)`.

`get_object/5`

Fetches an object into a local file path and returns its response metadata as a separate argument.

Compilation flags:

`static`

Template:

`get_object(Bucket,Key,File,Properties,Options)`

Mode and number of proofs:

`get_object(+atom,+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Bucket` is a variable:

`instantiation_error`

`Key` is a variable:

`instantiation_error`

`File` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

`File` is neither a variable nor an atom:

`type_error(atom,File)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`File` exists but is not a writable file:

`domain_error(file([],[write]),File)`

The directory `Directory` exists but is not writable:

`domain_error(directory([write]),Directory)`

The directory `Directory` does not exist:

`existence_error(directory,Directory)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid S3 client option:

`domain_error(s3_client_option,Option)`

The S3 service returned a non-success status:
`domain_error(s3_http_status,Status)`

Remarks:

- Options: Supported options are: `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))` for access credentials; `session_token(Token)` for temporary credentials; `region(Region)` for the AWS region, defaulting to `us-east-1` when `endpoint/1` is absent; `endpoint(URL)` for a custom S3-compatible endpoint; `addressing_style(virtual_hosted|path)` for endpoint addressing, defaulting to `virtual_hosted`; `headers(Headers)` for extra normalized request headers; `version(http(Major, Minor))` for the HTTP protocol version, defaulting to `http(1,1)`; `properties(Properties)` for extra normalized HTTP request properties; `connection_options(ConnectionOptions)` for transport-specific HTTP client options; `payload_hash_mode(signed|unsigned)` for payload signing, defaulting to `signed`; `request_time(date_time(...))` for an explicit local request time; `utc_offset_seconds(OffsetSeconds)` for deriving UTC from the local clock, defaulting to `0`; and `amz_date(AmzDate)` for an explicit AWS SigV4 timestamp overriding time derivation.

`put_object/6`

Uploads a local file path and returns the object ETag and response metadata as separate arguments.

Compilation flags:
`static`

Template:
`put_object(Bucket,Key,File,ETag,Properties,Options)`

Mode and number of proofs:
`put_object(+atom,+atom,+atom,-atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Bucket` is a variable:
`instantiation_error`
`Key` is a variable:
`instantiation_error`
`File` is a variable:
`instantiation_error`
`Bucket` is neither a variable nor an atom:
`type_error(atom,Bucket)`
`Key` is neither a variable nor an atom:
`type_error(atom,Key)`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`Bucket` is an atom but not a valid S3 bucket:
`domain_error(s3_bucket,Bucket)`
`File` exists but is not a readable file:
`domain_error(file([],[read]),File)`
The file `File` does not exist:
`existence_error(file,File)`
`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid S3 client option:

`domain_error(s3_client_option,Option)`

The S3 service returned a non-success status:

`domain_error(s3_http_status,Status)`

Remarks:

- Options: Supported options are the standard S3 request options: `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`, optional `session_token(Token)`, `region(Region)`, optional custom `endpoint(URL)`, `addressing_style(virtual_hosted|path)`, optional `headers(Headers)`, `version(http(Major, Minor))`, optional `properties(Properties)`, optional `connection_options(ConnectionOptions)`, `payload_hash_mode(signed|unsigned)`, optional `request_time(date_time(...))`, `utc_offset_seconds(OffsetSeconds)`, and optional explicit `amz_date(AmzDate)`.

`delete_object/4`

Deletes an object, returning `delete_result(Properties)`.

Compilation flags:

`static`

Template:

`delete_object(Bucket,Key,Result,Options)`

Mode and number of proofs:

`delete_object(+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Bucket` is a variable:

`instantiation_error`

`Key` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
`Options` contains an invalid S3 client option:
`domain_error(s3_client_option,Option)`
The S3 service returned a non-success status:
`domain_error(s3_http_status,Status)`

Remarks:

- Options: Supported options are the standard S3 request options: `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`, optional `session_token(Token)`, `region(Region)`, optional custom `endpoint(URL)`, `addressing_style(virtual_hosted|path)`, optional `headers(Headers)`, `version(http(Major, Minor))`, optional `properties(Properties)`, optional `connection_options(ConnectionOptions)`, `payload_hash_mode(signed|unsigned)`, optional `request_time(date_time(...))`, `utc_offset_seconds(OffsetSeconds)`, and optional explicit `amz_date(AmzDate)`.

`copy_object/5`

Copies an object from a source bucket/key into the destination bucket/key, returning `copy_result(ETag, Properties)`.

Compilation flags:
`static`

Template:
`copy_object(Source,Bucket,Key,Result,Options)`
Mode and number of proofs:
`copy_object(+compound,+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Bucket` is a variable:
`instantiation_error`
`Key` is a variable:
`instantiation_error`
The version identifier in `Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a compound term:
`type_error(compound,Source)`
`Bucket` is neither a variable nor an atom:
`type_error(atom,Bucket)`
`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

The version identifier in `Source` is neither a variable nor an atom:

`type_error(atom,VersionId)`

`Source` is a compound term but not a valid copy source term:

`domain_error(s3_copy_source,Source)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

The version identifier in `Source` is an atom but not a valid version identifier:

`domain_error(s3_version_id,VersionId)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid S3 client option:

`domain_error(s3_client_option,Option)`

The S3 service returned a non-success status or malformed XML body:

`domain_error(s3_http_status,Status)`

Remarks:

- Options: Supported options are the standard S3 request options: `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`, optional `session_token(Token)`, `region(Region)`, optional custom `endpoint(URL)`, `addressing_style(virtual_hosted|path)`, optional `headers(Headers)`, `version(http(Major, Minor))`, optional `properties(Properties)`, optional `connection_options(ConnectionOptions)`, `payload_hash_mode(signed|unsigned)`, optional `request_time(date_time(...))`, `utc_offset_seconds(OffsetSeconds)`, and optional explicit `amz_date(AmzDate)`.

`presigned_get_object/4`

Generates a presigned URL for downloading an object with an external HTTP client.

Compilation flags:

`static`

Template:

`presigned_get_object(Bucket,Key,URL,Options)`

Mode and number of proofs:

`presigned_get_object(+atom,+atom,-atom,+list(compound))` - `one_or_error`

Exceptions:

`Bucket` is a variable:

`instantiation_error`

`Key` is a variable:
```
instantiation_error
```
`Bucket` is neither a variable nor an atom:
```
type_error(atom,Bucket)
```
`Key` is neither a variable nor an atom:
```
type_error(atom,Key)
```
`Bucket` is an atom but not a valid S3 bucket:
```
domain_error(s3_bucket,Bucket)
```
`Options` is a variable:
```
instantiation_error
```
`Options` is neither a variable nor a list:
```
type_error(list,Options)
```
An element `Option` of the list `Options` is a variable:
```
instantiation_error
```
An element `Option` of the list `Options` is neither a variable nor a compound term:
```
type_error(compound,Option)
```
An element `Option` of the list `Options` is a compound term but not a valid option:
```
domain_error(option,Option)
```
`Options` contains an invalid S3 client option:
```
domain_error(s3_client_option,Option)
```
Credentials are missing or invalid:
```
domain_error(s3_credentials,missing)
```
The expiration is invalid or outside the AWS SigV4 range:
```
domain_error(s3_presign_expires,Expires)
```
The payload hash is missing, invalid, or incompatible with unsigned mode:
```
domain_error(s3_presign_payload_hash,Hash)
```

Remarks:

- Option `credentials/1`: Required access credentials as `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`.
- Option `session_token/1`: Optional session token for temporary credentials.
- Option `region/1`: AWS region to use when `endpoint/1` is absent. Defaults to `us-east-1`.
- Option `endpoint/1`: Optional custom S3-compatible endpoint URL.
- Option `addressing_style/1`: Endpoint addressing style, either `virtual_hosted` or `path`. Defaults to `virtual_hosted`.
- Option `headers/1`: Optional extra normalized request headers. When present they are signed into the URL and must be reproduced by the eventual caller.
- Option `payload_hash_mode/1`: Presigned URLs default to `payload_hash_mode(unsigned)`. Use `signed` together with `payload_hash/1` to sign a precomputed payload hash.
- Option `payload_hash/1`: Precomputed hexadecimal SHA-256 payload hash required when `payload_hash_mode(signed)` is selected.
- Option `request_time/1`: Optional explicit local request time as `date_time(Year, Month, Day, Hours, Minutes, Seconds)`.
- Option `utc_offset_seconds/1`: UTC offset used to derive the signing time from the local clock. Defaults to `0`.
- Option `amz_date/1`: Optional explicit AWS SigV4 timestamp overriding any derived request time.
- Option `expires/1`: Presigned URL lifetime in seconds. Defaults to `900` and must not exceed `604800`.

presigned_put_object/4

Generates a presigned URL for uploading an object with an external HTTP client.

Compilation flags:

`static`

Template:

`presigned_put_object(Bucket,Key,URL,Options)`

Mode and number of proofs:

`presigned_put_object(+atom,+atom,-atom,+list(compound))` - `one_or_error`

Exceptions:

`Bucket` is a variable:

`instantiation_error`

`Key` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid S3 client option:

`domain_error(s3_client_option,Option)`

Credentials are missing or invalid:

`domain_error(s3_credentials,missing)`

The expiration is invalid or outside the AWS SigV4 range:

`domain_error(s3_presign_expires,Expires)`

The payload hash is missing, invalid, or incompatible with unsigned mode:

`domain_error(s3_presign_payload_hash,Hash)`

Remarks:

- Option `credentials/1`: Required access credentials as `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`.
- Option `session_token/1`: Optional session token for temporary credentials.
- Option `region/1`: AWS region to use when `endpoint/1` is absent. Defaults to `us-east-1`.
- Option `endpoint/1`: Optional custom S3-compatible endpoint URL.
- Option `addressing_style/1`: Endpoint addressing style, either `virtual_hosted` or `path`. Defaults to `virtual_hosted`.

- Option `headers/1`: Optional extra normalized request headers. When present they are signed into the URL and must be reproduced by the eventual caller.
- Option `payload_hash_mode/1`: Presigned URLs default to `payload_hash_mode(unsigned)`. Use `signed` together with `payload_hash/1` to sign a precomputed payload hash.
- Option `payload_hash/1`: Precomputed hexadecimal SHA-256 payload hash required when `payload_hash_mode(signed)` is selected.
- Option `request_time/1`: Optional explicit local request time as `date_time(Year, Month, Day, Hours, Minutes, Seconds)`.
- Option `utc_offset_seconds/1`: UTC offset used to derive the signing time from the local clock. Defaults to `0`.
- Option `amz_date/1`: Optional explicit AWS SigV4 timestamp overriding any derived request time.
- Option `expires/1`: Presigned URL lifetime in seconds. Defaults to `900` and must not exceed `604800`.

`presigned_post_object/4`

Generates a presigned URL for a POST request to an object with an external HTTP client.

Compilation flags:

`static`

Template:

`presigned_post_object(Bucket,Key,URL,Options)`

Mode and number of proofs:

`presigned_post_object(+atom,+atom,-atom,+list(compound))` - `one_or_error`

Exceptions:

`Bucket` is a variable:

`instantiation_error`

`Key` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Options` contains an invalid S3 client option:

`domain_error(s3_client_option,Option)`

Credentials are missing or invalid:

`domain_error(s3_credentials,missing)`

The expiration is invalid or outside the AWS SigV4 range:

`domain_error(s3_presign_expires,Expires)`

The payload hash is missing, invalid, or incompatible with unsigned mode:

`domain_error(s3_presign_payload_hash,Hash)`

Remarks:

- Option `credentials/1`: Required access credentials as `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`.
- Option `session_token/1`: Optional session token for temporary credentials.
- Option `region/1`: AWS region to use when `endpoint/1` is absent. Defaults to `us-east-1`.
- Option `endpoint/1`: Optional custom S3-compatible endpoint URL.
- Option `addressing_style/1`: Endpoint addressing style, either `virtual_hosted` or `path`. Defaults to `virtual_hosted`.
- Option `headers/1`: Optional extra normalized request headers. When present they are signed into the URL and must be reproduced by the eventual caller.
- Option `payload_hash_mode/1`: Presigned URLs default to `payload_hash_mode(unsigned)`. Use `signed` together with `payload_hash/1` to sign a precomputed payload hash.
- Option `payload_hash/1`: Precomputed hexadecimal SHA-256 payload hash required when `payload_hash_mode(signed)` is selected.
- Option `request_time/1`: Optional explicit local request time as `date_time(Year, Month, Day, Hours, Minutes, Seconds)`.
- Option `utc_offset_seconds/1`: UTC offset used to derive the signing time from the local clock. Defaults to `0`.
- Option `amz_date/1`: Optional explicit AWS SigV4 timestamp overriding any derived request time.
- Option `expires/1`: Presigned URL lifetime in seconds. Defaults to `900` and must not exceed `604800`.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

object

1.288.6 `s3_session(HTTPTransport)`

- `HTTPTransport` - The object implementing `http_transport_protocol`.

Explicit S3 client sessions carrying default request options.

Availability:

`logtalk_load(s3(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-07-13

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*

Uses:
list
s3_client(HTTPTransport)
url(Representation)

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *get_object/6*
  - *open/1*
  - *open/2*
  - *close/1*
  - *list_buckets/3*
  - *head_bucket/4*
  - *list_objects_v2/5*
  - *head_object/5*
  - *presigned_get_object/5*
  - *presigned_put_object/5*
  - *presigned_post_object/5*
  - *put_object/7*
  - *delete_object/5*
  - *copy_object/6*
- *Protected predicates*
- *Private predicates*
  - *session_seed_/1*
  - *session_state_/2*
- *Operators*

Public predicates

get_object/6

Fetches an object into a local file path using session defaults merged with per-call options and returns its response metadata as a separate argument.

Compilation flags:

`static`

Template:

`get_object(Session,Bucket,Key,File,Properties,Options)`

Mode and number of proofs:

`get_object(+compound,+atom,+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Session` is a variable:

`instantiation_error`

`Session` is neither a variable nor a valid S3 session handle:

`domain_error(s3_session,Session)`

`Session` is a valid S3 session handle but not open:

`existence_error(s3_session,Session)`

`Bucket` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Key` is a variable:

`instantiation_error`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

`File` is a variable:

`instantiation_error`

`File` is neither a variable nor an atom:

`type_error(atom,File)`

`File` exists but is not a writable file:

`domain_error(file([],[write]),File)`

The directory `Directory` exists but is not writable:

`domain_error(directory([write]),Directory)`

The directory `Directory` does not exist:

`existence_error(directory,Directory)`

The delegated S3 client request failed:

`domain_error(s3_http_status,Status)`

Remarks:

- Option precedence: Per-call options are merged before the stored session defaults, so explicit call options take precedence.
- Option `credentials/1`: Access credentials as `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`.
- Option `session_token/1`: Optional session token for temporary credentials.

- Option `region/1`: AWS region to use when `endpoint/1` is absent.
- Option `endpoint/1`: Optional custom S3-compatible endpoint URL.
- Option `addressing_style/1`: Endpoint addressing style, either `virtual_hosted` or `path`.
- Option `headers/1`: Optional extra normalized HTTP request headers.
- Option `version/1`: HTTP protocol version as `http(Major, Minor)`.
- Option `properties/1`: Optional extra normalized HTTP request properties.
- Option `connection_options/1`: Optional transport-specific HTTP client options.
- Option `payload_hash_mode/1`: Payload signing mode, either `signed` or `unsigned`.
- Option `request_time/1`: Optional explicit local request time as `date_time(Year, Month, Day, Hours, Minutes, Seconds)`.
- Option `utc_offset_seconds/1`: UTC offset used to derive the signing time from the local clock.
- Option `amz_date/1`: Optional explicit AWS SigV4 timestamp overriding any derived request time.

`open/1`

Opens a new S3 client session with no default options.

Compilation flags:
`static`

Template:
`open(Session)`
Mode and number of proofs:
`open(-compound)` - `one_or_error`

Exceptions:

`open/2`

Opens a new S3 client session with the given default options.

Compilation flags:
`static`

Template:
`open(Session,Options)`
Mode and number of proofs:
`open(-compound,+list(compound))` - `one_or_error`

Exceptions:
`Options` contains an invalid S3 session option:
`domain_error(option,Option)`

Remarks:

- Option role: `Options` stores default request options reused by later session calls.

- Option `credentials/1`: Default access credentials as `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`.
- Option `session_token/1`: Optional default session token for temporary credentials.
- Option `region/1`: Default AWS region to use when `endpoint/1` is absent.
- Option `endpoint/1`: Optional default custom S3-compatible endpoint URL.
- Option `addressing_style/1`: Default endpoint addressing style, either `virtual_hosted` or `path`.
- Option `headers/1`: Optional default normalized HTTP request headers.
- Option `version/1`: Default HTTP protocol version as `http(Major, Minor)`.
- Option `properties/1`: Optional default normalized HTTP request properties.
- Option `connection_options/1`: Optional default transport-specific HTTP client options.
- Option `payload_hash_mode/1`: Default payload signing mode, either `signed` or `unsigned`.
- Option `request_time/1`: Optional default local request time as `date_time(Year, Month, Day, Hours, Minutes, Seconds)`.
- Option `utc_offset_seconds/1`: Default UTC offset used to derive the signing time from the local clock.
- Option `amz_date/1`: Optional default AWS SigV4 timestamp overriding any derived request time.

`close/1`

Closes a S3 client session.

Compilation flags:
`static`

Template:
`close(Session)`

Mode and number of proofs:
`close(+compound)` - `one_or_error`

Exceptions:
`Session` is a variable:
`instantiation_error`
`Session` is neither a variable nor a valid S3 session handle:
`domain_error(s3_session,Session)`
`Session` is a valid S3 session handle but not open:
`existence_error(s3_session,Session)`

`list_buckets/3`

Lists accessible buckets using session defaults merged with per-call options.

Compilation flags:
`static`

Template:
`list_buckets(Session,Buckets,Options)`

Mode and number of proofs:

```
list_buckets(+compound,-compound,+list(compound)) - one_or_error
```

Exceptions:

`Session` is a variable:

```
instantiation_error
```

`Session` is neither a variable nor a valid S3 session handle:

```
domain_error(s3_session,Session)
```

`Session` is a valid S3 session handle but not open:

```
existence_error(s3_session,Session)
```

`Options` contains an invalid S3 session option:

```
domain_error(option,Option)
```

The delegated S3 client request failed:

```
domain_error(s3_http_status,Status)
```

Remarks:

- Options: Per-call options accept the same terms as `open/2` and are merged before the stored session defaults, so explicit call options take precedence.

---

head_bucket/4

Returns bucket metadata using session defaults merged with per-call options.

Compilation flags:

```
static
```

Template:

```
head_bucket(Session,Bucket,Metadata,Options)
```

Mode and number of proofs:

```
head_bucket(+compound,+atom,-compound,+list(compound)) - one_or_error
```

Exceptions:

`Session` is a variable:

```
instantiation_error
```

`Session` is neither a variable nor a valid S3 session handle:

```
domain_error(s3_session,Session)
```

`Session` is a valid S3 session handle but not open:

```
existence_error(s3_session,Session)
```

`Bucket` is a variable:

```
instantiation_error
```

`Bucket` is neither a variable nor an atom:

```
type_error(atom,Bucket)
```

`Bucket` is an atom but not a valid S3 bucket:

```
domain_error(s3_bucket,Bucket)
```

The delegated S3 client request failed:

```
domain_error(s3_http_status,Status)
```

Remarks:

- Options: Per-call options accept the same terms as `open/2` and are merged before the stored session defaults, so explicit call options take precedence.

`list_objects_v2/5`

Lists objects using session defaults merged with per-call options.

Compilation flags:

`static`

Template:

`list_objects_v2(Session,Bucket,Request,Listing,Options)`

Mode and number of proofs:

`list_objects_v2(+compound,+atom,+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Session` is a variable:

`instantiation_error`

`Session` is neither a variable nor a valid S3 session handle:

`domain_error(s3_session,Session)`

`Session` is a valid S3 session handle but not open:

`existence_error(s3_session,Session)`

`Bucket` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Request` is a variable:

`instantiation_error`

An element `Option` of the request option list is a variable:

`instantiation_error`

An element `Option` of the request option list is neither a variable nor a compound term:

`type_error(compound,Option)`

`Request` is not a valid ListObjectsV2 request term or option list:

`domain_error(s3_list_objects_request,Request)`

An element `Option` of the request option list is a compound term but not a valid ListObjectsV2 request option:

`domain_error(s3_list_objects_request_option,Option)`

The delegated S3 client request failed:

`domain_error(s3_http_status,Status)`

Remarks:

- Options: Per-call options accept the same terms as `open/2` and are merged before the stored session defaults, so explicit call options take precedence. The `Request` argument carries the ListObjectsV2-specific request terms.

---

head_object/5

Returns object metadata using session defaults merged with per-call options.

Compilation flags:
`static`

Template:
`head_object(Session,Bucket,Key,Metadata,Options)`
Mode and number of proofs:
`head_object(+compound,+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Session` is a variable:
`instantiation_error`
`Session` is neither a variable nor a valid S3 session handle:
`domain_error(s3_session,Session)`
`Session` is a valid S3 session handle but not open:
`existence_error(s3_session,Session)`
`Bucket` is a variable:
`instantiation_error`
`Bucket` is neither a variable nor an atom:
`type_error(atom,Bucket)`
`Bucket` is an atom but not a valid S3 bucket:
`domain_error(s3_bucket,Bucket)`
`Key` is a variable:
`instantiation_error`
`Key` is neither a variable nor an atom:
`type_error(atom,Key)`
The delegated S3 client request failed:
`domain_error(s3_http_status,Status)`

Remarks:

- Options: Per-call options accept the same terms as `open/2` and are merged before the stored session defaults, so explicit call options take precedence.

---

presigned_get_object/5

Generates a presigned download URL using session defaults merged with per-call options.

Compilation flags:
`static`

Template:
`presigned_get_object(Session,Bucket,Key,URL,Options)`
Mode and number of proofs:
`presigned_get_object(+compound,+atom,+atom,-atom,+list(compound))` - `one_or_error`

Exceptions:

`Session` is a variable:

`instantiation_error`

`Session` is neither a variable nor a valid S3 session handle:

`domain_error(s3_session,Session)`

`Session` is a valid S3 session handle but not open:

`existence_error(s3_session,Session)`

`Bucket` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Key` is a variable:

`instantiation_error`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

The delegated S3 client request failed:

`domain_error(s3_client_option,Option)`

Remarks:

- Options: Per-call options accept the same terms as `open/2` plus `expires(Seconds)` for the URL lifetime and `payload_hash(Hash)` when `payload_hash_mode(signed)` is selected. Presigned URLs default to `payload_hash_mode(unsigned)`. Options are merged before the stored session defaults, so explicit call options take precedence.

`presigned_put_object/5`

Generates a presigned upload URL using session defaults merged with per-call options.

Compilation flags:

`static`

Template:

`presigned_put_object(Session,Bucket,Key,URL,Options)`

Mode and number of proofs:

`presigned_put_object(+compound,+atom,+atom,-atom,+list(compound))` - `one_or_error`

Exceptions:

`Session` is a variable:

`instantiation_error`

`Session` is neither a variable nor a valid S3 session handle:

`domain_error(s3_session,Session)`

`Session` is a valid S3 session handle but not open:

`existence_error(s3_session,Session)`

`Bucket` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Key` is a variable:

`instantiation_error`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

The delegated S3 client request failed:

`domain_error(s3_client_option,Option)`

Remarks:

- Options: Per-call options accept the same terms as `open/2` plus `expires(Seconds)` for the URL lifetime and `payload_hash(Hash)` when `payload_hash_mode(signed)` is selected. Presigned URLs default to `payload_hash_mode(unsigned)`. Options are merged before the stored session defaults, so explicit call options take precedence.

`presigned_post_object/5`

Generates a presigned POST URL using session defaults merged with per-call options.

Compilation flags:

`static`

Template:

`presigned_post_object(Session,Bucket,Key,URL,Options)`

Mode and number of proofs:

`presigned_post_object(+compound,+atom,+atom,-atom,+list(compound))` - `one_or_error`

Exceptions:

`Session` is a variable:

`instantiation_error`

`Session` is neither a variable nor a valid S3 session handle:

`domain_error(s3_session,Session)`

`Session` is a valid S3 session handle but not open:

`existence_error(s3_session,Session)`

`Bucket` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Key` is a variable:

`instantiation_error`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

The delegated S3 client request failed:

`domain_error(s3_client_option,Option)`

Remarks:

- Options: Per-call options accept the same terms as `open/2` plus `expires(Seconds)` for the URL lifetime and `payload_hash(Hash)` when `payload_hash_mode(signed)` is selected. Presigned URLs default to `payload_hash_mode(unsigned)`. Options are merged before the stored session defaults, so explicit call options take precedence.

`put_object/7`

Uploads a local file path using session defaults merged with per-call options and returns the object ETag and response metadata as separate arguments.

Compilation flags:

`static`

Template:

`put_object(Session,Bucket,Key,File,ETag,Properties,Options)`

Mode and number of proofs:

`put_object(+compound,+atom,+atom,+atom,-atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Session` is a variable:

`instantiation_error`

`Session` is neither a variable nor a valid S3 session handle:

`domain_error(s3_session,Session)`

`Session` is a valid S3 session handle but not open:

`existence_error(s3_session,Session)`

`Bucket` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Key` is a variable:

`instantiation_error`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

`File` is a variable:

`instantiation_error`

`File` is neither a variable nor an atom:

`type_error(atom,File)`

`File` exists but is not a readable file:

`domain_error(file([],[read]),File)`

The file `File` does not exist:

`existence_error(file,File)`

The delegated S3 client request failed:

`domain_error(s3_http_status,Status)`

Remarks:

- Options: Per-call options accept the same terms as `open/2` and are merged before the stored session defaults, so explicit call options take precedence.

`delete_object/5`

Deletes an object using session defaults merged with per-call options, returning `delete_result(Properties)`.

Compilation flags:

`static`

Template:

`delete_object(Session,Bucket,Key,Result,Options)`

Mode and number of proofs:

`delete_object(+compound,+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Session` is a variable:

`instantiation_error`

`Session` is neither a variable nor a valid S3 session handle:

`domain_error(s3_session,Session)`

`Session` is a valid S3 session handle but not open:

`existence_error(s3_session,Session)`

`Bucket` is a variable:

`instantiation_error`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

`Key` is a variable:

`instantiation_error`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

The delegated S3 client request failed:

`domain_error(s3_http_status,Status)`

Remarks:

- Options: Per-call options accept the same terms as `open/2` and are merged before the stored session defaults, so explicit call options take precedence.

---

copy_object/6

Copies an object using session defaults merged with per-call options, returning `copy_result(ETag, Properties)`.

Compilation flags:

`static`

Template:

`copy_object(Session,Source,Bucket,Key,Result,Options)`

Mode and number of proofs:

`copy_object(+compound,+compound,+atom,+atom,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Session` is a variable:

`instantiation_error`

`Session` is neither a variable nor a valid S3 session handle:

`domain_error(s3_session,Session)`

`Session` is a valid S3 session handle but not open:

`existence_error(s3_session,Session)`

`Bucket` is a variable:

`instantiation_error`

`Source` is a variable:

`instantiation_error`

The version identifier in `Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a compound term:

`type_error(compound,Source)`

`Bucket` is neither a variable nor an atom:

`type_error(atom,Bucket)`

The version identifier in `Source` is neither a variable nor an atom:

`type_error(atom,VersionId)`

`Source` is a compound term but not a valid copy source term:

`domain_error(s3_copy_source,Source)`

`Bucket` is an atom but not a valid S3 bucket:

`domain_error(s3_bucket,Bucket)`

The version identifier in `Source` is an atom but not a valid version identifier:

`domain_error(s3_version_id,VersionId)`

`Key` is a variable:

`instantiation_error`

`Key` is neither a variable nor an atom:

`type_error(atom,Key)`

The delegated S3 client request failed:

`domain_error(s3_http_status,Status)`

Remarks:

- Options: Per-call options accept the same terms as `open/2` and are merged before the stored session defaults, so explicit call options take precedence.

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

session_seed_/1

Dynamic counter used to generate fresh session identifiers.

Compilation flags:
`dynamic`

Template:
`session_seed_(SessionId)`
Mode and number of proofs:
`session_seed_(?positive_integer)` - `zero_or_one`

---

session_state_/2

Dynamic mapping between session identifiers and their stored default option lists.

Compilation flags:
`dynamic`

Template:
`session_state_(SessionId,Options)`
Mode and number of proofs:
`session_state_(?positive_integer,?list(compound))` - `zero_or_more`

Operators

(none)
object

1.288.7 s3_signing

AWS Signature Version 4 helpers specialized for S3.

Availability:
`logtalk_load(s3(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-09

Compilation flags:

`static, context_switching_calls`

Imports:
`public` *options*
Uses:
date
hmac
http_core
list
os
sha256
term_io
user

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *payload_hash/2*
  - *canonical_query/2*
  - *canonical_request/6*
  - *sign_request/8*
  - *presigned_query_pairs/7*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

payload_hash/2

Returns the hexadecimal SHA-256 payload hash for a normalized HTTP body term.

Compilation flags:
`static`

Template:
`payload_hash(Body,Hash)`
Mode and number of proofs:
`payload_hash(+compound,-atom)` - `one_or_error`

Exceptions:
`Body` is not a valid normalized HTTP body term:
`domain_error(http_body,Body)`

No registered HTTP body codec exists for the body media type when codec-based encoding is required:
`existence_error(http_body_codec,MediaType)`

---

canonical_query/2

Returns the canonical AWS SigV4 query string for the given name-value pairs.

Compilation flags:
`static`

Template:
`canonical_query(Pairs,Query)`

Mode and number of proofs:
`canonical_query(+list(pair),-atom)` - `one_or_error`

Exceptions:
`Pairs` is not a valid list of canonical query pairs:
`domain_error(s3_query_pairs,Pairs)`

---

canonical_request/6

Builds the canonical request string for the given method, URI, query pairs, signed headers, and payload hash.

Compilation flags:
`static`

Template:
`canonical_request(Method,CanonicalURI,QueryPairs,Headers,PayloadHash,CanonicalRequest)`

Mode and number of proofs:
`canonical_request(+atom,+atom,+list(pair),+list(compound),+atom,-atom)` - `one_or_error`

Exceptions:
`Method` is not a supported HTTP method for this helper:
`domain_error(s3_method,Method)`
`QueryPairs` is not a valid list of canonical query pairs:
`domain_error(s3_query_pairs,QueryPairs)`

---

sign_request/8

Signs an S3 request and returns the final normalized header list.

Compilation flags:

`static`

Template:

`sign_request(Method,HostHeader,CanonicalURI,QueryPairs,Body,Headers,SignedHeaders,Options)`

Mode and number of proofs:

`sign_request(+atom,+atom,+atom,+list(pair),+compound,+list(compound),-list(compound),+list(compound))` - `one_or_error`

Exceptions:

`Options` is missing valid credentials:

`domain_error(s3_credentials,missing)`

`Body` is not a valid normalized HTTP body term:

`domain_error(http_body,Body)`

`Headers` contains a managed or invalid signed header:

`domain_error(s3_managed_header,Header)`

The signing timestamp is invalid:

`domain_error(s3_amz_date,AmzDate)`

Remarks:

- Option `credentials/1`: Required access credentials as `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`.
- Option `session_token/1`: Optional session token for temporary credentials.
- Option `region/1`: AWS region for the credential scope. Defaults to `us-east-1`.
- Option `payload_hash_mode/1`: Payload signing mode, either `signed` or `unsigned`. Defaults to `signed`.
- Option `request_time/1`: Optional explicit local request time as `date_time(Year, Month, Day, Hours, Minutes, Seconds)`.
- Option `utc_offset_seconds/1`: UTC offset used to derive the signing time from the local clock. Defaults to `0`.
- Option `amz_date/1`: Optional explicit AWS SigV4 timestamp overriding any derived request time.

---

presigned_query_pairs/7

Signs an S3 request for query-string authentication and returns the final canonical query pairs, including the signature.

Compilation flags:

`static`

Template:

`presigned_query_pairs(Method,HostHeader,CanonicalURI,QueryPairs,Headers,SignedQueryPairs,Options)`

Mode and number of proofs:

`presigned_query_pairs(+atom,+atom,+atom,+list(pair),+list(compound),-list(pair),+list(compound))` - `one_or_error`

Exceptions:

`Options` is missing valid credentials:

`domain_error(s3_credentials,missing)`

`Headers` contains a managed or invalid signed header:

`domain_error(s3_managed_header,Header)`

The signing timestamp is invalid:

`domain_error(s3_amz_date,AmzDate)`

The expiration is invalid or outside the AWS SigV4 range:

`domain_error(s3_presign_expires,Expires)`

The payload hash is missing, invalid, or incompatible with unsigned mode:

`domain_error(s3_presign_payload_hash,Hash)`

Remarks:

- Option `credentials/1`: Required access credentials as `credentials(access_key_id(AccessKeyId), secret_access_key(SecretAccessKey))`.
- Option `session_token/1`: Optional session token for temporary credentials. It is emitted as the `X-Amz-Security-Token` query parameter.
- Option `region/1`: AWS region for the credential scope. Defaults to `us-east-1`.
- Option `payload_hash_mode/1`: Presigned URLs default to `unsigned` payload hashing and use `UNSIGNED-PAYLOAD`. Use `signed` together with `payload_hash/1` to sign a precomputed payload hash.
- Option `payload_hash/1`: Precomputed hexadecimal SHA-256 payload hash required when `payload_hash_mode(signed)` is selected.
- Option `request_time/1`: Optional explicit local request time as `date_time(Year, Month, Day, Hours, Minutes, Seconds)`.
- Option `utc_offset_seconds/1`: UTC offset used to derive the signing time from the local clock. Defaults to `0`.
- Option `amz_date/1`: Optional explicit AWS SigV4 timestamp overriding any derived request time.
- Option `expires/1`: Presigned URL lifetime in seconds. Defaults to `900` and must not exceed `604800`.

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.288.8 s3_xml

Adapter layer mapping S3 XML responses to stable library terms.

Availability:
`logtalk_load(s3(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-02

Compilation flags:
`static, context_switching_calls`

Uses:
list
user
xml

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *list_buckets_response/2*
  - *list_objects_v2_response/2*
  - *copy_object_response/2*
  - *create_multipart_upload_response/2*
  - *error_response/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

list_buckets_response/2

Parses a ListBuckets XML response body.

Compilation flags:
`static`

Template:
`list_buckets_response(Body,Buckets)`
Mode and number of proofs:
`list_buckets_response(+compound,-compound)` - `one_or_error`

Exceptions:

`Body` is not a supported XML response body term:

`domain_error(s3_xml_body,Body)`

The XML document is malformed or missing required elements:

`domain_error(s3_xml_document,Codes)`

`list_objects_v2_response/2`

Parses a ListObjectsV2 XML response body.

Compilation flags:

`static`

Template:

`list_objects_v2_response(Body,Listing)`

Mode and number of proofs:

`list_objects_v2_response(+compound,-compound)` - `one_or_error`

Exceptions:

`Body` is not a supported XML response body term:

`domain_error(s3_xml_body,Body)`

The XML document is malformed or missing required elements:

`domain_error(s3_xml_document,Codes)`

`copy_object_response/2`

Parses a CopyObject XML response body.

Compilation flags:

`static`

Template:

`copy_object_response(Body,Result)`

Mode and number of proofs:

`copy_object_response(+compound,-compound)` - `one_or_error`

Exceptions:

`Body` is not a supported XML response body term:

`domain_error(s3_xml_body,Body)`

The XML document is malformed or missing required elements:

`domain_error(s3_xml_document,Codes)`

---

create_multipart_upload_response/2

Parses a CreateMultipartUpload XML response body.

Compilation flags:

`static`

Template:

`create_multipart_upload_response(Body,Result)`

Mode and number of proofs:

`create_multipart_upload_response(+compound,-compound)` - `one_or_error`

Exceptions:

`Body` is not a supported XML response body term:

`domain_error(s3_xml_body,Body)`

The XML document is malformed or missing required elements:

`domain_error(s3_xml_document,Codes)`

---

error_response/2

Parses an S3 XML error response body.

Compilation flags:

`static`

Template:

`error_response(Body,Error)`

Mode and number of proofs:

`error_response(+compound,-compound)` - `zero_or_one_or_error`

Exceptions:

`Body` is not a supported XML response body term:

`domain_error(s3_xml_body,Body)`

The XML document is malformed or missing required elements:

`domain_error(s3_xml_document,Codes)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.289 sarif

object

1.289.1 `sarif`

Shared SARIF report generator for tools implementing the diagnostics protocol.

Availability:
`logtalk_load(sarif(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-10-23

Compilation flags:
`static, context_switching_calls`

Uses:
atom
git
json(ObjectRepresentation,PairRepresentation,StringRepresentation)
list
logtalk
os
reader
term_io
type
url(Representation)
user
uuid

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - `term/2`

    - *term/4*
    - *generate/2*
    - *generate/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

term/2

Returns a SARIF 2.1.0 report term for the given list of diagnostics tool specifications. Each specification must be a `tool_spec(Tool, Target, Options)` term.

Compilation flags:
`static`

Template:
`term(Specs,Term)`
Mode and number of proofs:
`term(+list(compound),-compound)` - `one`

---

term/4

Returns a SARIF 2.1.0 report term for a diagnostics tool target using the given options.

Compilation flags:
`static`

Template:
`term(Tool,Target,Term,Options)`
Mode and number of proofs:
`term(+object_identifier,+nonvar,-compound,+list(compound))` - `one`

---

generate/2

Generates a SARIF 2.1.0 report for the given list of diagnostics tool specifications and sink accepted by the json library. Each specification must be a `tool_spec(Tool, Target, Options)` term.

Compilation flags:
`static`

Template:
`generate(Specs,Sink)`

Mode and number of proofs:
`generate(+list(compound),++compound)` - `one`

`generate/4`

Generates a SARIF 2.1.0 report for a diagnostics tool target using the given options and sink accepted by the json library.

Compilation flags:
`static`

Template:
`generate(Tool,Target,Sink,Options)`
Mode and number of proofs:
`generate(+object_identifier,+nonvar,++compound,+list(compound))` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.290 sat

object

1.290.1 `sat`

Process-backed Kissat, CaDiCaL, Glucose, and Z3 CLI solving, DIMACS CNF source generation, and collected result inspection.

Availability:
`logtalk_load(sat(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-10

Compilation flags:
`static, context_switching_calls`

Implements:

public *sat_protocol*

Imports:

public *options*

Uses:

list
os
reader
sat_output
sat_process
sat_program
term_io
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *literal/2* *option/2* *option/3* *print_program/1* *print_result/1* *program/1* *program/2* *program/3* *result_property/2* *solve/2* *solve/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.290.2 sat_expansion

Opt-in equivalent CNF directive and clause expansion for pure problem objects.

Availability:
`logtalk_load(sat(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-10

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
sat_program

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.290.3 sat_output

Whole-document Kissat, CaDiCaL, Glucose, and Z3 plain-text output decoding.

Availability:
`logtalk_load(sat(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Uses:
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *decode/8*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

decode/8

Decodes captured stdout and process metadata into a collected `sat_result(Status, Model, Properties)` term, mapping reported literals back through an optional `Variables` table.

Compilation flags:
`static`

Template:
`decode(Solver,Codes,RawStatus,NormalizedStatus,Diagnostics,SourceMap,Variables,Result)`
Mode and number of proofs:

`decode(+atom,+list(code),+term,+compound,+list(code),+list(compound),+term,-compound)` - `one_or_error`

Exceptions:

`Solver` is not a ground term:
`instantiation_error`
`Solver` is neither a variable nor an atom:
`type_error(atom,Solver)`
`Codes` is a partial list or a list with an element `Element` which is a variable:
`instantiation_error`
`Codes` is neither a partial list nor a list of character codes:
`type_error(codes,Codes)`
`Codes` has no status line or an unsupported status label:
`domain_error(sat_output,Field-Value)`
The witness line is missing its zero terminator:
`domain_error(sat_output,witness_terminator_missing)`
A witness literal number has no entry in the variable table:
`domain_error(sat_output,variable(Number))`
`RawStatus` is not a ground term:
`instantiation_error`
`NormalizedStatus` is not a ground term:
`instantiation_error`
`NormalizedStatus` is fatal or inconsistent with the decoded solver result:
`domain_error(sat_process_status,NormalizedStatus)`
`Diagnostics` is a partial list or a list with an element `Element` which is a variable:
`instantiation_error`
`Diagnostics` is neither a partial list nor a list of character codes:
`type_error(codes,Diagnostics)`
`SourceMap` is a partial list or a list with an element `Element` which is a variable:
`instantiation_error`
`SourceMap` is neither a variable nor a list:
`type_error(list,SourceMap)`
`Variables` is not a ground term:
`instantiation_error`
`Variables` is ground but neither `none` nor a list of pairs:
`type_error(list(pair),Variables)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.290.4 `sat_problem_protocol`

Protocol for ordered, ground CNF statement providers.

Availability:
`logtalk_load(sat(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-10

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *cnf_statement/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`cnf_statement/2`

Enumerates, by backtracking, canonical CNF statements in source order and their origins.

Compilation flags:
`static`

Template:
`cnf_statement(Statement,Origin)`
Mode and number of proofs:
`cnf_statement(-compound,-term)` - `zero_or_more`

Remarks:

- Origin: Either `source(File, Start, End)` or `unknown`.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.290.5 `sat_process`

Isolated operating-system shell transport for the Kissat, CaDiCaL, Glucose, and Z3 CLIs.

Availability:
`logtalk_load(sat(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Uses:
list
os
process
reader

type
user
utf_8

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *quote/3*
  - *run/7*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

quote/3

Quotes an atom as one POSIX shell or Windows PowerShell literal, without interpolation.

Compilation flags:
`static`

Template:
`quote(Platform,Argument,Quoted)`
Mode and number of proofs:
`quote(+atom,+atom,-atom)` - `one_or_error`

Exceptions:
`Platform` is a variable:
`instantiation_error`
`Platform` is neither a variable nor an atom:
`type_error(atom,Platform)`
`Platform` is an atom but neither `posix` nor `windows`:
`domain_error(sat_shell_platform,Platform)`
`Argument` is a variable:
`instantiation_error`
`Argument` is neither a variable nor an atom:
`type_error(atom,Argument)`
`Argument` is an atom but contains a null character:
`domain_error(sat_shell_argument,Argument)`

run/7

Runs the CLI using an owned directory and separate UTF-8 output files, returning only after cleanup.

Compilation flags:

`static`

Template:

`run(Executable,Arguments,Source,Codes,Diagnostics,RawStatus,NormalizedStatus)`

Mode and number of proofs:

`run(+atom,+list(atom),+compound,-list(code),-list(code),-term,-compound)` - `one_or_error`

Exceptions:

`Executable` is a variable:

`instantiation_error`

`Executable` is neither a variable nor an atom:

`type_error(atom,Executable)`

`Arguments` is a partial list or a list with an element `Argument` which is a variable:

`instantiation_error`

`Arguments` is neither a variable nor a list:

`type_error(list(atom),Arguments)`

An element `Argument` of the list `Arguments` is neither a variable nor an atom:

`type_error(atom,Argument)`

`Executable` or an element `Argument` of the list `Arguments` contains a null character:

`domain_error(sat_shell_argument,Argument)`

`Source` is not a ground term:

`instantiation_error`

`Source` is neither `file(Path)` nor `codes(Codes)`:

`domain_error(sat_process_source,Source)`

A launcher dependency does not exist:

`existence_error(command,Command)`

A source or capture file does not exist:

`existence_error(source_sink,Path)`

A source or capture file cannot be opened:

`permission_error(open,source_sink,Path)`

An owned directory cannot be found during cleanup:

`existence_error(directory,Directory)`

An owned file or directory cannot be removed:

`permission_error(delete,source_sink,Path)`

A character sequence is not valid UTF-8:

`representation_error(utf_8)`

A launcher operation fails:

`sat_transport(Stage,Status,Details)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.290.6 `sat_program`

Canonical CNF statement validation, DIMACS source generation, and variable table computation.

Availability:
`logtalk_load(sat(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-10

Compilation flags:
`static, context_switching_calls`

Uses:
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *serialize/2*
  - *serialize/4*
  - *variables/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

serialize/2

Serializes ground canonical CNF statements to DIMACS character codes, assigning sequential variable numbers by first occurrence.

Compilation flags:
`static`

Template:
`serialize(Statements,Codes)`
Mode and number of proofs:
`serialize(+list,-list(code))` - `one_or_error`

Exceptions:
`Statements` is a partial list or a list with an element `Statement` which is a variable:
`instantiation_error`
`Statements` is neither a partial list nor a list:
`type_error(list,Statements)`
An element `Statement` of the list `Statements` is not a valid statement:
`domain_error(sat_statement,Statement)`

---

serialize/4

Serializes statements with origins and returns `line(Start, End, Origin)` source-map entries; the generated header occupies line 1.

Compilation flags:
`static`

Template:
`serialize(Statements,Origins,Codes,SourceMap)`
Mode and number of proofs:
`serialize(+list,+list,-list(code),-list(compound))` - `one_or_error`

Exceptions:
`Statements` is a partial list or a list with an element `Statement` which is a variable:
`instantiation_error`
`Statements` is neither a partial list nor a list:
`type_error(list,Statements)`
An element `Statement` of the list `Statements` is not a valid statement:
`domain_error(sat_statement,Statement)`
`Origins` is a partial list or a list with an element `Origin` which is a variable:
`instantiation_error`
`Origins` is neither a variable nor a list:
`type_error(list,Origins)`
`Origins` has an element that is neither `unknown` nor a valid `source(File, Start, End)` term, or its length differs from `Statements`:

`domain_error(sat_origins,Origins)`

`variables/2`

Returns the ordered `Term-Number` variable table that `serialize/2` and `serialize/4` assign by first occurrence across clause statements.

Compilation flags:
`static`

Template:
`variables(Statements,Variables)`
Mode and number of proofs:
`variables(+list,-list(pair))` - `one_or_error`

Exceptions:
`Statements` is a partial list or a list with an element `Statement` which is a variable:
`instantiation_error`
`Statements` is neither a partial list nor a list:
`type_error(list,Statements)`
An element `Statement` of the list `Statements` is not a valid statement:
`domain_error(sat_statement,Statement)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.290.7 `sat_protocol`

Protocol for process-backed Kissat, CaDiCaL, Glucose, and Z3 CLI solving, source generation, and result inspection.

Availability:
`logtalk_load(sat(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *solve/2*
  - *solve/3*
  - *program/1*
  - *program/2*
  - *program/3*
  - *print_program/1*
  - *literal/2*
  - *result_property/2*
  - *print_result/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

solve/2

Solves a source with default options using the external SAT solver CLI and returns a collected result after cleanup. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`.

Compilation flags:
`static`

Template:
`solve(Source,Result)`

Mode and number of proofs:
`solve(+compound,-compound)` - `one_or_error`

Exceptions:
`Source` is not a ground term:
`instantiation_error`
`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `sat_problem_protocol`:
`domain_error(sat_source,Source)`
`Statements` is ground but not a list:

`type_error(list,Statements)`
A statement enumerated by the provider object is not a valid statement:
`domain_error(sat_statement,Statement)`
An origin enumerated by the provider object is invalid, or its origins and statements differ in length:
`domain_error(sat_origins,Origins)`
`Codes` is ground but not a list of character codes:
`type_error(codes,Codes)`
An element `Code` of the list `Codes` is ground but not an integer:
`type_error(integer,Code)`
An element `Code` of the list `Codes` is an integer but not a valid character code:
`domain_error(character_code,Code)`
`Path` is ground but not an atom:
`type_error(atom,Path)`
`Path` does not exist:
`existence_error(source_sink,Path)`
`Path` cannot be opened for reading:
`permission_error(open,source_sink,Path)`
A launcher argument contains a null character:
`domain_error(sat_shell_argument,Argument)`
The selected solver executable or a shell dependency does not exist:
`existence_error(command,Command)`
An owned directory cannot be found during cleanup:
`existence_error(directory,Directory)`
An owned file or directory cannot be removed:
`permission_error(delete,source_sink,Path)`
A character sequence is not valid UTF-8:
`representation_error(utf_8)`
A launcher operation fails:
`sat_transport(Stage,Status,Details)`
Solver output is invalid or inconsistent with its exit status:
`sat_solver_error(Formal,Status,Diagnostics,SourceMap)`

`solve/3`

Solves a source using `solver/1`, `executable/1`, `time_limit/1`, `conflicts/1`, `decisions/1`, `seed/1`, and `witness/1` options. Repeated options use the first occurrence. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`.

Compilation flags:
`static`

Template:
`solve(Source,Result,Options)`
Mode and number of proofs:
`solve(+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Source` is not a ground term:
    `instantiation_error`
`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `sat_problem_protocol`:
    `domain_error(sat_source,Source)`
`Statements` is ground but not a list:
    `type_error(list,Statements)`
A statement enumerated by the provider object is not a valid statement:
    `domain_error(sat_statement,Statement)`
An origin enumerated by the provider object is invalid, or its origins and statements differ in length:
    `domain_error(sat_origins,Origins)`
`Codes` is ground but not a list of character codes:
    `type_error(codes,Codes)`
An element `Code` of the list `Codes` is ground but not an integer:
    `type_error(integer,Code)`
An element `Code` of the list `Codes` is an integer but not a valid character code:
    `domain_error(character_code,Code)`
`Path` is ground but not an atom:
    `type_error(atom,Path)`
`Path` does not exist:
    `existence_error(source_sink,Path)`
`Path` cannot be opened for reading:
    `permission_error(open,source_sink,Path)`
`Options` is a variable:
    `instantiation_error`
`Options` is neither a variable nor a list:
    `type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
    `instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
    `type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
    `domain_error(option,Option)`
A launcher argument contains a null character:
    `domain_error(sat_shell_argument,Argument)`
The selected solver executable or a shell dependency does not exist:
    `existence_error(command,Command)`
An owned directory cannot be found during cleanup:
    `existence_error(directory,Directory)`
An owned file or directory cannot be removed:
    `permission_error(delete,source_sink,Path)`
A character sequence is not valid UTF-8:
    `representation_error(utf_8)`
A launcher operation fails:
    `sat_transport(Stage,Status,Details)`
Solver output is invalid or inconsistent with its exit status:
    `sat_solver_error(Formal,Status,Diagnostics,SourceMap)`

program/1

Generates and prints DIMACS CNF source to the current output using default options, equivalent to `program/2` followed by `print_program/1`, without invoking the solver.

Compilation flags:

`static`

Template:

`program(Source)`

Mode and number of proofs:

`program(+compound)` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `sat_problem_protocol`:

`domain_error(sat_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(sat_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(sat_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

The current output stream does not allow writing:

`permission_error(output,stream,Stream)`

program/2

Returns DIMACS CNF source codes from `object(Object)`, `terms(Statements)`, `codes(Codes)`, or `file(Path)` using default options, without invoking the solver.

Compilation flags:

`static`

Template:

`program(Source,Codes)`

Mode and number of proofs:

`program(+compound,-list(integer))` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `sat_problem_protocol`:

`domain_error(sat_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(sat_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(sat_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

---

program/3

Returns DIMACS CNF source codes from `object(Object)`, `terms(Statements)`, `codes(Codes)`, or `file(Path)`, using the `embed_variables/1` option, without invoking the solver.

Compilation flags:

`static`

Template:

`program(Source,Codes,Options)`

Mode and number of proofs:

`program(+compound,-list(integer),+list(compound))` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `sat_problem_protocol`:

`domain_error(sat_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(sat_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(sat_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

print_program/1

Prints DIMACS CNF character codes to the current output without adding a newline or invoking the solver.

Compilation flags:

`static`

Template:

`print_program(Codes)`

Mode and number of proofs:

`print_program(+list(integer))` - `one_or_error`

Exceptions:

`Codes` is not a ground term:

`instantiation_error`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

The current output stream does not allow writing:

`permission_error(output,stream,Stream)`

---

literal/2

Enumerates, by backtracking, the collected model literals (`+Term` or `-Term`) without starting a process. Fails when the result has no witness.

Compilation flags:

`static`

Template:

`literal(Result,Literal)`

Mode and number of proofs:

`literal(+compound,-compound)` - `zero_or_more`

Exceptions:

`Result` is not a ground term:

`instantiation_error`

`Result` is not a valid collected SAT solver result:

`domain_error(sat_result,Result)`

---

result_property/2

Enumerates, by backtracking, `status(Status)` and the stored result properties.

Compilation flags:

`static`

Template:

`result_property(Result,Property)`

Mode and number of proofs:

`result_property(+compound,?compound)` - `zero_or_more`

Exceptions:

`Result` is not a ground term:

`instantiation_error`

`Result` is not a valid collected SAT solver result:

`domain_error(sat_result,Result)`

---

print_result/1

Prints the collected status and, when present, the witness literals to the current output using Kissat-style `s`/`v` labels, without invoking the solver.

Compilation flags:

`static`

Template:

`print_result(Result)`

Mode and number of proofs:

`print_result(+compound)` - `one_or_error`

Exceptions:

`Result` is not a ground term:

`instantiation_error`

`Result` is not a valid collected SAT solver result:

`domain_error(sat_result,Result)`

The current output stream does not allow writing:

`permission_error(output,stream,Stream)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.291 sbom

object

1.291.1 `sbom`

This tool generates a Software Bill of Materials (SBOM) for an application.

Availability:
`logtalk_load(sbom(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2026-11-16

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
json
json_schema
list
os
packs
term_io
url(Representation)
user
uuid

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *document/2*
  - *document/1*
  - *export/2*
  - *export/1*
- *Protected predicates*
- *Private predicates*
  - *spdx_license_schema_/1*
- *Operators*

Public predicates

document/2

Returns an SPDX 2.3 or a CycloneDX 1.7 JSON term describing the currently loaded application using the given options. The JSON term represents objects using curly terms, pairs using a dash, and strings using atoms.

Compilation flags:

```
static
```

Template:

```
document(Document,Options)
```

Mode and number of proofs:

```
document(-compound,+list(compound)) - one
```

---

document/1

Returns an SPDX 2.3 JSON term describing the currently loaded application using default options. The JSON term represents objects using curly terms, pairs using a dash, and strings using atoms.

Compilation flags:

```
static
```

Template:

```
document(Document)
```

Mode and number of proofs:

```
document(-compound) - one
```

---

export/2

Exports an SPDX 2.3 or a CycloneDX 1.7 JSON document describing the currently loaded application to the specified sink using the given options. Valid sinks are `codes(List)`, `stream(Stream)`, `file(Path)`, `chars(List)`, and `atom(Atom)`.

Compilation flags:

`static`

Template:

`export(Sink,Options)`

Mode and number of proofs:

`export(++compound,+list(compound))` - `one`

---

export/1

Exports an SPDX 2.3 JSON document describing the currently loaded application to the specified sink using default options. Valid sinks are `codes(List)`, `stream(Stream)`, `file(Path)`, `chars(List)`, and `atom(Atom)`.

Compilation flags:

`static`

Template:

`export(Sink)`

Mode and number of proofs:

`export(++compound)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

spdx_license_schema_/1

Caches the parsed schema used to validate SPDX license identifiers for CycloneDX exports.

Compilation flags:

`dynamic`

Template:

`spdx_license_schema_(Schema)`

Mode and number of proofs:

`spdx_license_schema_(-term)` - `zero_or_one`

Operators

(none)

1.292 schulze_ranker

object

1.292.1 `schulze_ranker`

Schulze pairwise preference ranker. Learns one deterministic score per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol by computing strongest paths over aggregated head-to-head outcomes and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:
`logtalk_load(schulze_ranker(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-07

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ranking_dataset_common*
`public` *score_ranker_model_common*
`public` *condorcet_victory_common*

Uses:
avltree
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
  - *strongest_paths/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

strongest_paths/2

Returns the labeled strongest-path relation as `path(Item1,Item2,Strength)` terms for all ordered pairs of distinct learned items.

Compilation flags:

```
static
```

Template:

```
strongest_paths(Ranker,StrongestPaths)
```

Mode and number of proofs:

```
strongest_paths(+compound,-list(compound)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *copeland_ranker*, *borda_ranker*

1.293 sequential_pattern_mining_protocols

protocol

1.293.1 sequence_dataset_protocol

Protocol for ordered sequence datasets used with sequential pattern mining algorithms. Sequence identifiers are expected to be unique within a dataset.

Availability:
`logtalk_load(sequential_pattern_mining_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-29

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *items/1*
  - *sequence/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

items/1

Returns the declared sequence item domain as a canonical sorted list of unique items.

Compilation flags:
`static`

Template:
`items(Items)`
Mode and number of proofs:
`items(-list(atom))` - `one`

sequence/2

Enumerates by backtracking the sequences in the dataset. The sequence identifier is expected to be unique within the dataset. Each sequence is represented as an ordered list of canonical sorted itemsets with unique declared items.

Compilation flags:
`static`

Template:
`sequence(Id,Sequence)`
Mode and number of proofs:
`sequence(-integer,-list(list(atom)))` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
category

1.293.2 sequential_pattern_mining_common

Shared predicates for sequential pattern miner dataset validation, support counting, and pattern ordering helpers.

Availability:
`logtalk_load(sequential_pattern_mining_protocols(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-08-02

Compilation flags:
`static`

Extends:
`public` *pattern_miner_common*
Uses:
list
type

Remarks:
(none)

Inherited public predicates:

check_option/1 check_options/1 check_pattern_miner/1 default_option/1 default_options/1 diagnostic/2 diagnostics/2 export_to_clauses/4 export_to_file/4 mine/2 mine/3 option/2 option/3 pattern_miner_options/2 print_pattern_miner/1 valid_option/1 valid_options/1 valid_pattern_miner/1

- *Public predicates*
- *Protected predicates*
  - *check_sequences/4*
  - *pattern_length/2*
  - *sort_patterns/2*
  - *filter_patterns/3*
  - *count_items/4*
  - *select_frequent_item_supports/3*
  - *sort_item_supports/2*
  - *valid_sequence_patterns/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

check_sequences/4

Validates the collected dataset sequences and returns the maximum total sequence length.

Compilation flags:

```
static
```

Template:

```
check_sequences(Dataset,ItemDomain,Sequences,MaxSequenceLength)
```

Mode and number of proofs:

```
check_sequences(+object_identifier,+list(atom),+list(pair(integer,
list(list(atom)))),-integer) - one
```

---

pattern_length/2

Computes the total number of items in a sequential pattern.

Compilation flags:

```
static
```

Template:

```
pattern_length(Pattern,PatternLength)
```

Mode and number of proofs:

```
pattern_length(+list(list(atom)),-integer) - one
```

---

sort_patterns/2

Sorts sequence patterns first by total item count and then lexicographically.

Compilation flags:

```
static
```

Template:

```
sort_patterns(Patterns0,Patterns)
```

Mode and number of proofs:

```
sort_patterns(+list(compound),-list(compound)) - one
```

---

filter_patterns/3

Filters sequence patterns by minimum total item count.

Compilation flags:

```
static
```

Template:

```
filter_patterns(Patterns0,MinimumPatternLength,Patterns)
```

Mode and number of proofs:

```
filter_patterns(+list(compound),+integer,-list(compound)) - one
```

---

count_items/4

Accumulates support counts for a list of items.

Compilation flags:

```
static
```

Template:

```
count_items(Items,Count,ItemCounts0,ItemCounts)
```

Mode and number of proofs:

```
count_items(+list(atom),+integer,+list(pair(atom,integer)),-list(pair(atom,
integer))) - one
```

---

select_frequent_item_supports/3

Selects the item supports that satisfy the minimum support count.

Compilation flags:
```
static
```

Template:
```
select_frequent_item_supports(ItemCounts,SupportCount,FrequentItemSupports)
```
Mode and number of proofs:
```
select_frequent_item_supports(+list(pair(atom,integer)),+integer,-list(compound))
- one
```

---

sort_item_supports/2

Sorts item supports lexicographically by item.

Compilation flags:
```
static
```

Template:
```
sort_item_supports(ItemSupports0,ItemSupports)
```
Mode and number of proofs:
```
sort_item_supports(+list(compound),-list(compound)) - one
```

---

valid_sequence_patterns/2

True when the patterns are valid `sequence_pattern(Pattern, Support)` terms over the given item domain with positive integer supports.

Compilation flags:
```
static
```

Template:
```
valid_sequence_patterns(ItemDomain,Patterns)
```
Mode and number of proofs:
```
valid_sequence_patterns(+list(atom),+list(compound)) - zero_or_one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.294 sets

object

1.294.1 `set`

Set predicates implemented using ordered lists. Uses `==/2` for element comparison and standard term ordering.

Availability:
`logtalk_load(sets(loader))`

Author: Richard O'Keefe (main predicates); adapted to Logtalk by Paulo Moura.
Version: 2:0:0
Date: 2025-07-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *setp*
Extends:
`public` *compound*
Aliases:
`setp` `size/2` as `length/2`

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_list/2 as_set/2 check/1 delete/3 depth/2 disjoint/2 empty/1 equal/2 ground/1 insert/3 insert_all/3 intersect/2 intersection/3 intersection/4 member/2 memberchk/2 new/1 numbervars/1 numbervars/3 occurs/2 powerset/2 product/3 select/3 selectchk/3 singletons/2 size/2 subset/2 subsumes/2 subterm/2 subtract/3 symdiff/3 union/3 union/4 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

set(Type), *treap_set*

object

1.294.2 `set(Type)`

Set predicates with elements constrained to a single type and custom comparing rules.

Availability:
`logtalk_load(sets(loader))`

Author: Paulo Moura and Adrian Arroyo
Version: 1:24:0
Date: 2022-02-03

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *set*
Uses:
list

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_list/2 as_set/2 check/1 delete/3 depth/2 disjoint/2 empty/1 equal/2 ground/1 insert/3 insert_all/3 intersect/2 intersection/3 intersection/4 member/2 memberchk/2 new/1 numbervars/1 numbervars/3 occurs/2 powerset/2 product/3 select/3 selectchk/3 singletons/2 size/2 subset/2 subsumes/2 subterm/2 subtract/3 symdiff/3 union/3 union/4 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *sort/2*
  - *partition/4*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

sort/2

Sorts a list in ascending order.

Compilation flags:
```
static
```

Template:
```
sort(List,Sorted)
```
Mode and number of proofs:
```
sort(+list,-list) - one
```

---

partition/4

List partition in two sub-lists using a pivot.

Compilation flags:
```
static
```

Template:
```
partition(List,Pivot,Lowers,Biggers)
```
Mode and number of proofs:
```
partition(+list,+nonvar,-list,-list) - one
```

Operators

(none)

See also

set, *treap_set*

protocol

1.294.3 `setp`

Set protocol.

Availability:
`logtalk_load(sets(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2025-07-08

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *as_set/2*
  - *as_list/2*
  - *delete/3*
  - *disjoint/2*
  - *equal/2*
  - *empty/1*
  - *insert/3*
  - *insert_all/3*
  - *intersect/2*
  - *intersection/3*
  - *intersection/4*
  - *size/2*
  - *member/2*
  - *memberchk/2*

      - *powerset/2*
      - *product/3*
      - *select/3*
      - *selectchk/3*
      - *subset/2*
      - *subtract/3*
      - *symdiff/3*
      - *union/3*
      - *union/4*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

as_set/2

Returns a set with all unique elements from the given list.

Compilation flags:
```
static
```

Template:
```
as_set(List,Set)
```
Mode and number of proofs:
```
as_set(@list,-set) - one
```

---

as_list/2

Returns a list with all elements of the given set.

Compilation flags:
```
static
```

Template:
```
as_list(Set,List)
```
Mode and number of proofs:
```
as_list(@set,-list) - one
```

---

delete/3

Deletes an element from a set returning the set of the remaining elements.

Compilation flags:
`static`

Template:
`delete(Set,Element,Remaining)`
Mode and number of proofs:
`delete(@set,@term,-set)` - `one`

---

disjoint/2

True when the two sets have no element in common.

Compilation flags:
`static`

Template:
`disjoint(Set1,Set2)`
Mode and number of proofs:
`disjoint(@set,@set)` - `zero_or_one`

---

equal/2

True when the two sets are equal.

Compilation flags:
`static`

Template:
`equal(Set1,Set2)`
Mode and number of proofs:
`equal(@set,@set)` - `zero_or_one`

---

empty/1

True when the set is empty.

Compilation flags:
```
static
```

Template:
```
empty(Set)
```
Mode and number of proofs:
```
empty(@set) - zero_or_one
```

---

insert/3

Inserts an element in a set, returning the resulting set.

Compilation flags:
```
static
```

Template:
```
insert(In,Element,Out)
```
Mode and number of proofs:
```
insert(@set,@term,-set) - one
```

---

insert_all/3

Inserts all the elements of the list into a set, returning the resulting set.

Compilation flags:
```
static
```

Template:
```
insert_all(List,In,Out)
```
Mode and number of proofs:
```
insert_all(@list,@set,-set) - one
```

---

intersect/2

True if the two sets have at least one element in common.

Compilation flags:
`static`

Template:
`intersect(Set1,Set2)`
Mode and number of proofs:
`intersect(@set,@set)` - `zero_or_one`

---

intersection/3

Computes the intersection of `Set1` and `Set2`.

Compilation flags:
`static`

Template:
`intersection(Set1,Set2,Intersection)`
Mode and number of proofs:
`intersection(@set,@set,-set)` - `one`

---

intersection/4

Computes the intersection and the difference between `Set2` and `Set1`.

Compilation flags:
`static`

Template:
`intersection(Set1,Set2,Intersection,Difference)`
Mode and number of proofs:
`intersection(@set,@set,-set,-set)` - `one`

---

size/2

Number of set elements.

Compilation flags:
`static`

Template:
`size(Set,Size)`
Mode and number of proofs:
`size(@set,?integer)` - `zero_or_one`

---

member/2

`Element` is a member of set `Set`.

Compilation flags:
`static`

Template:
`member(Element,Set)`
Mode and number of proofs:
`member(?term,+set)` - `zero_or_more`

---

memberchk/2

True when a term is a member of a set.

Compilation flags:
`static`

Template:
`memberchk(Element,Set)`
Mode and number of proofs:
`memberchk(@term,@set)` - `zero_or_one`

---

powerset/2

Returns the power set of a set, represented as a list of sets.

Compilation flags:
```
static
```

Template:
```
powerset(Set,Powerset)
```
Mode and number of proofs:
```
powerset(@set,-list(set)) - one
```

---

product/3

Returns the cartesian product of two sets.

Compilation flags:
```
static
```

Template:
```
product(Set1,Set2,Product)
```
Mode and number of proofs:
```
product(@set,@set,-set) - one
```

---

select/3

Selects an element from a set, returning the set of remaining elements.

Compilation flags:
```
static
```

Template:
```
select(Element,Set,Remaining)
```
Mode and number of proofs:
```
select(?term,?set,?set) - zero_or_more
```

---

selectchk/3

True if an element can be selected from a set, returning the set of remaining elements.

Compilation flags:
`static`

Template:
`selectchk(Element,Set,Remaining)`
Mode and number of proofs:
`selectchk(@term,@set,-set)` - `zero_or_one`

---

subset/2

True if `Subset` is a subset of `Set`.

Compilation flags:
`static`

Template:
`subset(Subset,Set)`
Mode and number of proofs:
`subset(@set,@set)` - `zero_or_one`

---

subtract/3

Computes the set of all the elements of `Set1` which are not also in `Set2`.

Compilation flags:
`static`

Template:
`subtract(Set1,Set2,Difference)`
Mode and number of proofs:
`subtract(@set,@set,-set)` - `one`

---

symdiff/3

Computes the symmetric difference of `Set1` and `Set2`, containing all elements that are not in the sets intersection.

Compilation flags:
`static`

Template:
`symdiff(Set1,Set2,Difference)`
Mode and number of proofs:
`symdiff(@set,@set,-set)` - `one`

---

union/3

Computes the union of `Set1` and `Set2`.

Compilation flags:
`static`

Template:
`union(Set1,Set2,Union)`
Mode and number of proofs:
`union(@set,@set,-set)` - `one`

---

union/4

Computes the union of `Set1` and `Set2` and the difference between `Set2` and `Set1`.

Compilation flags:
`static`

Template:
`union(Set1,Set2,Union,Difference)`
Mode and number of proofs:
`union(@set,@set,-set,-set)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

set, *set(Type)*

object

1.294.4 `treap_set`

Set predicates implemented using treaps (tree heaps). A treap is a binary search tree with randomly assigned priorities. Uses `==/2` for element comparison and standard term ordering.

Availability:
`logtalk_load(sets(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-01-28

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *setp*
Extends:
`public` *compound*
Uses:
fast_random(Algorithm)
list
set

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 as_list/2 as_set/2 check/1 delete/3 depth/2 disjoint/2 empty/1 equal/2 ground/1 insert/3 insert_all/3 intersect/2 intersection/3 intersection/4 member/2 memberchk/2 new/1 numbervars/1 numbervars/3 occurs/2 powerset/2 product/3 select/3 selectchk/3 singletons/2 size/2 subset/2 subsumes/2 subterm/2 subtract/3 symdiff/3 union/3 union/4 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

set, *set(Type)*

1.295 sgd_classifier

object

1.295.1 `sgd_classifier`

Stochastic gradient descent classifier supporting one-vs-rest linear models with configurable losses including `log_loss`, `hinge`, `squared_hinge`, `modified_huber`, and `perceptron`.

Availability:

```
logtalk_load(sgd_classifier(loader))
```

Author: Paulo Moura
Version: 1:0:2
Date: 2026-10-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *probabilistic_classifier_common*

Uses:

format
linear_algebra

list
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_classifier/1 *check_option/1* *check_options/1* *classifier_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict_probabilities/3* *print_classifier/1* *valid_classifier/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

dataset_protocol, *logistic_regression_classifier*, *linear_svm_classifier*, *knn_classifier*, *nearest_centroid_classifier*

1.296 simple_temporal_networks

object

1.296.1 stn

Immutable Simple Temporal Networks with retained source constraints and cached all-pairs bounds.

Availability:
`logtalk_load(simple_temporal_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-18

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stn_protocol*
Uses:
interval
list
type

Remarks:
(none)

Inherited public predicates:
add_constraint/5 *add_constraint/6* *add_constraints/3* *add_constraints/4* *add_time_points/3* *bounds/4* *can_precede/3* *can_precede_or_equal/3* *consistent/1* *constraints/2* *difference_bounds/5* *distance/4* *distance/5* *earliest/3* *earliest_schedule/2* *entails/4* *entails/5* *event_interval/4* *event_relation/6* *latest/3* *must_precede/3* *must_precede_or_equal/3* *new/2* *remove_constraints/3* *remove_time_points/3* *schedule/2* *time_points/2* *try_add_constraints/4* *window_interval/3* *window_relation/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

interval, *interval_constraint_network*

protocol

1.296.2 `stn_protocol`

Portable, functional Simple Temporal Networks over closed difference constraints.

Availability:
`logtalk_load(simple_temporal_networks(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-01

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- Constraints: A constraint from `X` to `Y` with weight `Delta` means `time(Y) - time(X) =< Delta`.
- Inputs: States must be opaque ground terms returned by this API. Malformed public arguments and unknown time-points fail without instantiating inputs; hand-built internal states are unsupported.
- Numbers: Weights are finite integers or floats. Arithmetic errors propagate; floating-point results are approximate, with no implicit tolerance.
- Bounds: Unbounded query results use the atoms positive_infinity and negative_infinity, never native numeric infinities.

Inherited public predicates:
(none)

- *Public predicates*
  - *new/2*
  - *add_time_points/3*
  - *remove_time_points/3*
  - *time_points/2*
  - *constraints/2*
  - *consistent/1*
  - *schedule/2*
  - *earliest_schedule/2*
  - *add_constraint/5*
  - *add_constraint/6*
  - *add_constraints/3*
  - *add_constraints/4*
  - *try_add_constraints/4*
  - *remove_constraints/3*
  - *distance/4*
  - *distance/5*
  - *entails/4*
  - *entails/5*
  - *earliest/3*
  - *latest/3*
  - *bounds/4*
  - *difference_bounds/5*
  - *can_precede/3*
  - *must_precede/3*
  - *can_precede_or_equal/3*
  - *must_precede_or_equal/3*
  - *window_interval/3*
  - *event_interval/4*
  - *window_relation/4*
  - *event_relation/6*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new/2

Creates an unconstrained network over distinct ground time-points and a reserved reference time-point zero fixed at time zero. Fails for duplicate labels or an explicitly supplied zero.

Compilation flags:

```
static
```

Template:

```
new(TimePoints,STN)
```

Mode and number of proofs:

```
new(+list,-stn) - zero_or_one
```

---

add_time_points/3

Appends distinct fresh ground time-points, rebuilding once and preserving source identifiers. Fails for existing labels or zero. An empty list returns the original state.

Compilation flags:

```
static
```

Template:

```
add_time_points(STN0,TimePoints,STN)
```

Mode and number of proofs:

```
add_time_points(+stn,+list,-stn) - zero_or_one_or_error
```

Exceptions:

Arithmetic cannot be evaluated:

```
evaluation_error(A)
```

Arithmetic exceeds a backend representation limit:

```
representation_error(A)
```

---

remove_time_points/3

Removes time-points and all incident source constraints, rebuilding once and preserving surviving node order and source identifiers. Missing labels and duplicate requests are ignored. Fails for a request containing zero or a malformed or nonground list.

Compilation flags:

```
static
```

Template:

```
remove_time_points(STN0,TimePoints,STN)
```

Mode and number of proofs:

```
remove_time_points(+stn,+list,-stn) - zero_or_one_or_error
```

Exceptions:

Arithmetic cannot be evaluated:

```
evaluation_error(A)
```

Arithmetic exceeds a backend representation limit:

```
representation_error(A)
```

Remarks:

- Bounds: Deletion retracts incident sources, not just variables. Bounds supported through removed points may weaken or become unbounded; implied bounds are not preserved as new constraints.
- Identity: Labels are compared by term identity. The original state and next source identifier are preserved. A request removing no existing point returns the original state.

---

time_points/2

Returns the declared time-points in insertion order, with zero first.

Compilation flags:
`static`

Template:
`time_points(STN,TimePoints)`
Mode and number of proofs:
`time_points(+stn,-list)` - `zero_or_one`

---

constraints/2

Returns all source constraints in insertion order as `constraint(Id, X, Y, Delta)` terms, including redundant and parallel constraints.

Compilation flags:
`static`

Template:
`constraints(STN,Constraints)`
Mode and number of proofs:
`constraints(+stn,-list(compound))` - `zero_or_one`

---

consistent/1

True if the argument is a library-owned consistent state. All successfully returned states are consistent; this predicate is not an importer for hand-built internal terms.

Compilation flags:
`static`

Template:
`consistent(STN)`
Mode and number of proofs:
`consistent(+stn)` - `zero_or_one`

---

schedule/2

Returns one finite feasible assignment as time(Point, Value) terms in declaration order, with time(zero, 0) first. Fails for invalid states without instantiating inputs.

Compilation flags:

```
static
```

Template:

```
schedule(STN,Schedule)
```

Mode and number of proofs:

```
schedule(+stn,-list(compound)) - zero_or_one_or_error
```

Exceptions:

Arithmetic cannot be evaluated:

```
evaluation_error(A)
```

Arithmetic exceeds a backend representation limit:

```
representation_error(A)
```

Remarks:

- Choice: The assignment is deterministic, not necessarily unique, earliest, non-negative, or optimal. Disconnected and absolutely unbounded points are assigned finite values.
- Numbers: Every retained source inequality is checked after normalization using backend arithmetic. Floating-point results remain approximate, with no implicit tolerance.

---

earliest_schedule/2

Returns the componentwise earliest feasible assignment as time(Point, Value) terms in declaration order, with time(zero, 0) first. Requires every declared point to have a finite absolute lower bound. Fails for invalid states or any unbounded lower bound without instantiating inputs.

Compilation flags:

```
static
```

Template:

```
earliest_schedule(STN,Schedule)
```

Mode and number of proofs:

```
earliest_schedule(+stn,-list(compound)) - zero_or_one_or_error
```

Exceptions:

Arithmetic cannot be evaluated:

```
evaluation_error(A)
```

Arithmetic exceeds a backend representation limit:

```
representation_error(A)
```

Remarks:

- Bounds: Finite upper bounds are not required. No artificial lower bounds are introduced and the original state and source identifiers are unchanged.

- Numbers: Componentwise minimality holds in exact arithmetic. Every retained source inequality is checked using backend arithmetic; floating-point feasibility and optimality are not exact-real guarantees and no tolerance is applied.

---

add_constraint/5

Adds `Y - X =< Delta`, retaining the source even when redundant. Fails if the result is inconsistent or the inputs are invalid.

Compilation flags:

```
static
```

Template:

```
add_constraint(STN0,X,Y,Delta,STN)
```

Mode and number of proofs:

```
add_constraint(+stn,@term,@term,+number,-stn) - zero_or_one_or_error
```

Exceptions:

Arithmetic cannot be evaluated:

```
evaluation_error(A)
```

Arithmetic exceeds a backend representation limit:

```
representation_error(A)
```

---

add_constraint/6

Adds `Y - X =< Delta` and returns its source identifier. Fails for an inconsistent result or invalid inputs. Identifiers are local to a state lineage, not globally unique across forks.

Compilation flags:

```
static
```

Template:

```
add_constraint(STN0,X,Y,Delta,Id,STN)
```

Mode and number of proofs:

```
add_constraint(+stn,@term,@term,+number,-integer,-stn) - zero_or_one_or_error
```

Exceptions:

Arithmetic cannot be evaluated:

```
evaluation_error(A)
```

Arithmetic exceeds a backend representation limit:

```
representation_error(A)
```

---

add_constraints/3

Atomically adds a batch of `constraint(X, Y, Delta)` terms, rebuilding once. Fails if any input is invalid or the combined network is inconsistent. An empty batch returns the original state.

Compilation flags:

`static`

Template:

`add_constraints(STN0,Constraints,STN)`

Mode and number of proofs:

`add_constraints(+stn,+list(compound),-stn)` - `zero_or_one_or_error`

Exceptions:

Arithmetic cannot be evaluated:

`evaluation_error(A)`

Arithmetic exceeds a backend representation limit:

`representation_error(A)`

---

add_constraints/4

Atomically adds `constraint(X, Y, Delta)` terms and returns their source identifiers in batch order. Fails for invalid inputs or an inconsistent result.

Compilation flags:

`static`

Template:

`add_constraints(STN0,Constraints,Ids,STN)`

Mode and number of proofs:

`add_constraints(+stn,+list(compound),-list(integer),-stn)` - `zero_or_one_or_error`

Exceptions:

Arithmetic cannot be evaluated:

`evaluation_error(A)`

Arithmetic exceeds a backend representation limit:

`representation_error(A)`

---

try_add_constraints/4

Attempts a batch update and returns `consistent(STN)` or `inconsistent(cycle(Sources, Weight))`. The cycle is an ordered closed path of identified source constraints with negative total weight, not necessarily a minimal conflict. Invalid inputs fail.

Compilation flags:

`static`

Template:

`try_add_constraints(STN0,Constraints,Ids,Outcome)`

Mode and number of proofs:

`try_add_constraints(+stn,+list(compound),-list(integer),-compound)` - `zero_or_one_or_error`

Exceptions:

Arithmetic cannot be evaluated:

`evaluation_error(A)`

Arithmetic exceeds a backend representation limit:

`representation_error(A)`

Remarks:

- Identifiers: Ids identifies candidate constraints in batch order even when rejected. No state is changed; rejected identifiers may coincide with later accepted allocations from the original state.
- Arithmetic: Backend arithmetic errors propagate. Non-finite computed sums raise evaluation_error(float_overflow); inconsistent numerical diagnostics raise evaluation_error(stn_numerical_inconsistency).

---

remove_constraints/3

Removes source constraints by positive integer identifier, rebuilding once if necessary. Missing identifiers and duplicate requests are ignored. Surviving identifiers are preserved and removed identifiers are never reused in descendant states.

Compilation flags:

`static`

Template:

`remove_constraints(STN0,Ids,STN)`

Mode and number of proofs:

`remove_constraints(+stn,+list(integer),-stn)` - `zero_or_one_or_error`

Exceptions:

Arithmetic cannot be evaluated:

`evaluation_error(A)`

Arithmetic exceeds a backend representation limit:

`representation_error(A)`

---

distance/4

Returns the tightest upper bound on `Y - X`, or positive_infinity if no directed path exists.

Compilation flags:

```
static
```

Template:

```
distance(STN,X,Y,Upper)
```

Mode and number of proofs:

```
distance(+stn,@term,@term,-bound) - zero_or_one
```

---

distance/5

Returns a finite tight upper bound on `Y - X` and one ordered supporting source-constraint path. Fails for an unreachable pair. An identity query returns zero and an empty path. Ties select one deterministic witness, not every support.

Compilation flags:

```
static
```

Template:

```
distance(STN,X,Y,Upper,Path)
```

Mode and number of proofs:

```
distance(+stn,@term,@term,-number,-list(compound)) - zero_or_one_or_error
```

Exceptions:

A numerical witness cannot be reconstructed:

```
evaluation_error(stn_numerical_inconsistency)
```

---

entails/4

True if the network entails `Y - X =< Delta` for a finite integer or float Delta.

Compilation flags:

```
static
```

Template:

```
entails(STN,X,Y,Delta)
```

Mode and number of proofs:

```
entails(+stn,@term,@term,+number) - zero_or_one
```

---

entails/5

True if the network entails `Y - X =< Delta` and returns one supporting source-constraint path.

Compilation flags:
`static`

Template:
`entails(STN,X,Y,Delta,Path)`
Mode and number of proofs:
`entails(+stn,@term,@term,+number,-list(compound))` - `zero_or_one`

Exceptions:
A numerical witness cannot be reconstructed:
`evaluation_error(stn_numerical_inconsistency)`

---

earliest/3

Returns the earliest feasible time relative to zero, or negative_infinity when unbounded.

Compilation flags:
`static`

Template:
`earliest(STN,TimePoint,Earliest)`
Mode and number of proofs:
`earliest(+stn,@term,-bound)` - `zero_or_one`

---

latest/3

Returns the latest feasible time relative to zero, or positive_infinity when unbounded.

Compilation flags:
`static`

Template:
`latest(STN,TimePoint,Latest)`
Mode and number of proofs:
`latest(+stn,@term,-bound)` - `zero_or_one`

---

bounds/4

Returns the tightest absolute time window, including symbolic unbounded endpoints. The reference point zero has bounds `[0, 0]`.

Compilation flags:

```
static
```

Template:

```
bounds(STN,TimePoint,Earliest,Latest)
```

Mode and number of proofs:

```
bounds(+stn,@term,-bound,-bound) - zero_or_one
```

---

difference_bounds/5

Returns tight bounds `Lower =< Y - X =< Upper`. `Lower` is the negated reverse distance, `Upper` the forward distance. Either endpoint may be symbolically unbounded.

Compilation flags:

```
static
```

Template:

```
difference_bounds(STN,X,Y,Lower,Upper)
```

Mode and number of proofs:

```
difference_bounds(+stn,@term,@term,-bound,-bound) - zero_or_one
```

---

can_precede/3

True if some feasible schedule has `X` strictly earlier than `Y`. Tests whether the upper bound on `Y - X` is positive or unbounded.

Compilation flags:

```
static
```

Template:

```
can_precede(STN,X,Y)
```

Mode and number of proofs:

```
can_precede(+stn,@term,@term) - zero_or_one
```

---

must_precede/3

True if every feasible schedule has `X` strictly earlier than `Y`. Tests whether the lower bound on `Y - X` is positive.

Compilation flags:
`static`

Template:
`must_precede(STN,X,Y)`
Mode and number of proofs:
`must_precede(+stn,@term,@term)` - `zero_or_one`

---

can_precede_or_equal/3

True if some feasible schedule has `X` no later than `Y`. Tests whether the upper bound on `Y - X` is non-negative or unbounded.

Compilation flags:
`static`

Template:
`can_precede_or_equal(STN,X,Y)`
Mode and number of proofs:
`can_precede_or_equal(+stn,@term,@term)` - `zero_or_one`

---

must_precede_or_equal/3

True if every feasible schedule has `X` no later than `Y`. Tests whether the lower bound on `Y - X` is non-negative.

Compilation flags:
`static`

Template:
`must_precede_or_equal(STN,X,Y)`
Mode and number of proofs:
`must_precede_or_equal(+stn,@term,@term)` - `zero_or_one`

---

window_interval/3

Returns a finite non-degenerate marginal time-point window as i(Start, End), preserving endpoint numeric representations. Fails for unknown points, invalid states, and unbounded or singleton windows.

Compilation flags:

```
static
```

Template:

```
window_interval(STN,TimePoint,Interval)
```

Mode and number of proofs:

```
window_interval(+stn,@term,-compound) - zero_or_one
```

Remarks:

- Numeric ordering: The descriptor uses arithmetic endpoint ordering, not term ordering. It is numeric data, not a portable replacement for the ranked interval bridge.
- Envelope: Describes marginal absolute bounds, not an event duration or joint endpoint correlations.

---

event_interval/4

Returns a concrete event interval as i(Start, End), preserving endpoint numeric representations. Both points must have finite singleton absolute bounds and the duration must be strictly positive. Fails for invalid states and unknown, uncertain, unbounded, reversed, or zero-duration endpoints.

Compilation flags:

```
static
```

Template:

```
event_interval(STN,StartPoint,EndPoint,Interval)
```

Mode and number of proofs:

```
event_interval(+stn,@term,@term,-compound) - zero_or_one
```

Remarks:

- Numeric ordering: The descriptor uses arithmetic endpoint ordering, not term ordering. Use event_relation/6 for the portable ranked Allen-relation bridge.

---

window_relation/4

Classifies two finite, non-degenerate marginal time-point windows using Allen relations. Fails for unbounded or singleton windows. This is an envelope diagnostic, not an entailed relation between uncertain events.

Compilation flags:

```
static
```

Template:

```
window_relation(STN,TimePoint1,TimePoint2,Relation)
```

Mode and number of proofs:

`window_relation(+stn,@term,@term,-atom)` - `zero_or_one`

Remarks:

- Numeric ordering: Delegates to interval after jointly ranking endpoints by arithmetic value, preserving numeric equality across integers and floats.

`event_relation/6`

Classifies two concrete event intervals using Allen relations. All four endpoint time-points must have finite singleton absolute bounds and each event must have strictly positive duration. Fails for uncertain, unbounded, reversed, or zero-duration events.

Compilation flags:

`static`

Template:

`event_relation(STN,Start1,End1,Start2,End2,Relation)`

Mode and number of proofs:

`event_relation(+stn,@term,@term,@term,@term,-atom)` - `zero_or_one`

Remarks:

- Numeric ordering: Delegates to interval after jointly ranking endpoints by arithmetic value, without converting integer endpoints to floats.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.297 simulated_annealing

object

1.297.1 `simulated_annealing(Problem)`

- `Problem` - Problem object implementing `simulated_annealing_protocol`.

Simulated annealing optimization algorithm using the Xoshiro128++ random number generator via `fast_random(xoshiro128pp)`.

Availability:

`logtalk_load(simulated_annealing(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:

`static, context_switching_calls`

Extends:

`public` *simulated_annealing(Problem,fast_random(xoshiro128pp))*

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *estimate_temperature/1* *estimate_temperature/2* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

simulated_annealing(Problem,Random), *simulated_annealing_protocol*

object

1.297.2 simulated_annealing(Problem,Random)

- `Problem` - Problem object implementing `simulated_annealing_protocol`.
- `Random` - Random number generator object providing `between/3`, `random/1`, and `randomize/1` predicates.

Simulated annealing optimization algorithm. Parameterized by a problem object implementing the `simulated_annealing_protocol` protocol and by a random number generator algorithm for the `fast_random` library. The algorithm minimizes the energy (cost) function defined by the problem. Custom cooling schedules, stop conditions, delta-energy neighbor generation, progress reporting, and reheating restarts can be defined by the problem object or configured via options; suitable defaults are used otherwise.

Availability:
`logtalk_load(simulated_annealing(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
type

Remarks:

- Algorithm: Simulated annealing is a probabilistic metaheuristic for global optimization. It explores the solution space by iteratively generating neighbor states and accepting them based on an energy-dependent probability that decreases over time as the temperature cools.
- Acceptance criterion: Uses the standard Boltzmann acceptance criterion: a worse neighbor is accepted with probability `exp(-DeltaE / Temperature)`.
- Default cooling schedule: Geometric cooling: `NewTemp is Temp * 0.995`. Override by defining `cooling_schedule/3` in the problem object.
- Default stop condition: The search stops when the maximum number of steps is reached or the temperature drops below the minimum temperature. Override by defining `stop_condition/3` in the problem object.
- Delta-energy optimization: If the problem object defines `neighbor_state/3`, the algorithm uses the returned delta energy directly instead of calling `state_energy/2` on the neighbor. This is useful when computing the energy change is cheaper than recomputing the full energy.
- Progress reporting: If the problem object defines `progress/5`, it is called periodically with the current step, temperature, best energy, acceptance rate, and improvement rate. The reporting interval is controlled by the `updates(N)` option. A final report is always produced when the loop terminates.

- Best state tracking: The algorithm tracks the best state found across all iterations and across all restart cycles, not just the final state.
- Seed control: The `seed(S)` option initializes the random number generator for reproducible runs.
- Reheating restart: The `restarts(N)` option runs N additional SA cycles after the first. Each restart reheats the temperature to the initial value and begins from the best state found so far, allowing the search to escape local minima. Statistics accumulate across all cycles.
- Auto-temperature estimation: The `estimate_temperature/1-2` predicates sample random neighbor transitions and compute an initial temperature that would produce a target acceptance rate. This avoids manual tuning of the initial temperature.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *estimate_temperature/1*
  - *estimate_temperature/2*
  - *run/2*
  - *run/3*
  - *run/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

estimate_temperature/1

Estimates an initial temperature for the problem using default options (200 samples, 80% target acceptance rate).

Compilation flags:

`static`

Template:

`estimate_temperature(Temperature)`

Mode and number of proofs:

`estimate_temperature(-float)` - `one`

---

estimate_temperature/2

Estimates an initial temperature by sampling random neighbor transitions. The temperature is computed so that the Boltzmann acceptance probability for the average uphill move equals the target acceptance rate. Sampling starts from the problem initial state and follows a random walk.

Compilation flags:

`static`

Template:

```
estimate_temperature(Temperature,Options)
```

Mode and number of proofs:

```
estimate_temperature(-float,+list(compound)) - one
```

Remarks:

- `samples(N)` option: Number of random neighbor transitions to sample (default: `200`).
- `acceptance_rate(P)` option: Target initial acceptance rate as an integer percentage between 1 and 99 (default: `80`).

---

run/2

Runs the simulated annealing algorithm using default options and returns the best state found and its energy.

Compilation flags:

```
static
```

Template:

```
run(BestState,BestEnergy)
```

Mode and number of proofs:

```
run(-term,-number) - one
```

---

run/3

Runs the simulated annealing algorithm using the given options and returns the best state found and its energy.

Compilation flags:

```
static
```

Template:

```
run(BestState,BestEnergy,Options)
```

Mode and number of proofs:

```
run(-term,-number,+list(compound)) - one
```

Remarks:

- `max_steps(N)` option: Maximum number of iterations per cycle (default: `10000`).
- `min_temperature(T)` option: Minimum temperature floor; search stops when temperature drops below this value (default: `0.001`).
- `updates(N)` option: Number of progress reports during the run. Set to `0` to disable. Progress is reported by calling `progress/5` on the problem object (default: `0`).
- `seed(S)` option: Positive integer seed for the random number generator, enabling reproducible runs (default: none).
- `restarts(N)` option: Number of additional SA cycles after the first. Each restart reheats the temperature and begins from the best state found so far (default: `0`).

`run/4`

Runs the simulated annealing algorithm using the given options, returns the best state found and its energy, and returns run statistics.

Compilation flags:
`static`

Template:
`run(BestState,BestEnergy,Statistics,Options)`
Mode and number of proofs:
`run(-term,-number,-list(compound),+list(compound))` - `one`

Remarks:

- Statistics list: A list of `Key(Value)` pairs: `steps(N)` is the number of steps executed, `acceptances(A)` is the number of accepted moves, `improvements(I)` is the number of moves that improved the best energy, and `final_temperature(T)` is the temperature at termination.

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

simulated_annealing(Problem), *simulated_annealing_protocol*

protocol

1.297.3 `simulated_annealing_protocol`

Protocol for simulated annealing problem definitions. A problem object must define the four required predicates and may optionally define predicates to override cooling, stopping, neighbor generation with delta energy, and progress reporting defaults.

Availability:
`logtalk_load(simulated_annealing(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-22

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *initial_state/1*
  - *neighbor_state/2*
  - *neighbor_state/3*
  - *state_energy/2*
  - *initial_temperature/1*
  - *cooling_schedule/3*
  - *stop_condition/3*
  - *progress/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

initial_state/1

Returns an initial state for the optimization problem.

Compilation flags:
`static`

Template:
`initial_state(State)`

Mode and number of proofs:
`initial_state(-term)` - `one`

---

neighbor_state/2

Generates a neighboring state from the given state. This is the most problem-specific predicate and its definition determines the quality of the search.

Compilation flags:
`static`

Template:

```
neighbor_state(State,Neighbor)
```

Mode and number of proofs:

```
neighbor_state(+term,-term) - one
```

---

neighbor_state/3

Generates a neighboring state and returns the energy change (delta) directly, avoiding a full energy recomputation. Optional. When not defined by the problem, the algorithm calls `neighbor_state/2` and `state_energy/2` instead.

Compilation flags:

```
static
```

Template:

```
neighbor_state(State,Neighbor,DeltaEnergy)
```

Mode and number of proofs:

```
neighbor_state(+term,-term,-number) - one
```

---

state_energy/2

Computes the energy (cost) of the given state. The algorithm minimizes this value.

Compilation flags:

```
static
```

Template:

```
state_energy(State,Energy)
```

Mode and number of proofs:

```
state_energy(+term,-number) - one
```

---

initial_temperature/1

Returns the initial temperature for the annealing schedule. Higher temperatures increase the probability of accepting worse solutions early on.

Compilation flags:

```
static
```

Template:

```
initial_temperature(Temperature)
```

Mode and number of proofs:

```
initial_temperature(-number) - one
```

---

cooling_schedule/3

Computes the next temperature from the current temperature and step number. Optional. When not defined by the problem, a default geometric cooling schedule is used (`NewTemp is Temp * 0.995`).

Compilation flags:

```
static
```

Template:

```
cooling_schedule(Temperature,Step,NewTemperature)
```

Mode and number of proofs:

```
cooling_schedule(+number,+non_negative_integer,-number) - one
```

---

stop_condition/3

True when the search should stop given the current step, temperature, and best energy found so far. Optional. When not defined by the problem, the search runs until the maximum number of steps is reached or the minimum temperature is reached.

Compilation flags:

```
static
```

Template:

```
stop_condition(Step,Temperature,BestEnergy)
```

Mode and number of proofs:

```
stop_condition(+non_negative_integer,+number,+number) - zero_or_one
```

---

progress/5

Called periodically to report optimization progress. Optional. When not defined by the problem, progress reporting is skipped. The acceptance and improvement rates are values between 0.0 and 1.0 computed over the interval since the last progress report.

Compilation flags:

```
static
```

Template:

```
progress(Step,Temperature,BestEnergy,AcceptanceRate,ImprovementRate)
```

Mode and number of proofs:

```
progress(+non_negative_integer,+number,+number,+number,+number) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

simulated_annealing(Problem)

1.298 slope_one_recommender

object

1.298.1 `slope_one_recommender`

Weighted Slope One recommender with canonical item-pair deviations and co-rating support counts.

Availability:
`logtalk_load(slope_one_recommender(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-04

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *recommender_common*

Uses:
list
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_recommender/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_recommender/1* *recommend/4* *recommender_options/2* *score/4* *valid_option/1* *valid_options/1* *valid_recommender/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

recommender_protocol

1.299 smt

object

1.299.1 `smt`

Process-backed Z3, cvc5, Bitwuzla, and Yices 2 CLI solving, SMT-LIB 2 source generation, and collected result inspection.

Availability:
`logtalk_load(smt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *smt_protocol*
Imports:
`public` *options*
Uses:
list

os
reader
smt_output
smt_process
smt_program
type

Remarks:
(none)

Inherited public predicates:
capability/2 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *model_binding/2* *model_value/3* *option/2* *option/3* *print_program/1* *print_result/1* *print_results/1* *program/1* *program/2* *result_property/2* *solve/2* *solve/3* *status/2* *unsat_core_name/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.299.2 `smt_expansion`

Opt-in statement expansion for pure problem objects, converting canonical statement facts and the `raw/1` and `comment/1` directives into `smt_statement/2` facts.

Availability:
`logtalk_load(smt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-08

Compilation flags:

`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
smt_program

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.299.3 `smt_output`

Whole-document S-expression tokenizing, parsing, and content-based response classification for Z3, cvc5, Bitwuzla, and Yices 2 plain-text output decoding.

Availability:
`logtalk_load(smt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-14

Compilation flags:
`static, context_switching_calls`

Uses:

list

type

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - `decode/7`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`decode/7`

Decodes captured stdout and process metadata into a list of collected `smt_result(Status, Model, Properties)` terms, one per check response, including objective responses when present.

Compilation flags:

`static`

Template:

`decode(Solver,Codes,RawStatus,NormalizedStatus,Diagnostics,SourceMap,Results)`

Mode and number of proofs:

`decode(+atom,+list(code),+term,+compound,+list(code),+list(compound),-list(compound))` - `one_or_error`

Exceptions:

`Solver` is not a ground term:

`instantiation_error`

`Solver` is neither a variable nor an atom:

`type_error(atom,Solver)`

`Codes` is a partial list or a list with an element `Element` which is a variable:

`instantiation_error`

`Codes` is neither a partial list nor a list of character codes:

`type_error(codes,Codes)`

`Codes` contains an unrecognized character or an unbalanced parenthesis:

`domain_error(smt_output,Field-Value)`

A model, unsat core, or get-value response appears before any status response:

`domain_error(smt_output,Response-unexpected)`

The solver reported an `(error ...)` response:

`smt_solver_reported_error(Message)`

The solver reported an unsupported command:

`smt_solver_reported_error(unsupported)`

`RawStatus` is not a ground term:

`instantiation_error`

`NormalizedStatus` is not a ground term:

`instantiation_error`

`NormalizedStatus` is neither a variable nor one of `exit(Code)`, `unknown(Details)`, nor `killed(Details)`:

`domain_error(smt_process_status,NormalizedStatus)`

No response was parsed and the process did not exit cleanly:

`domain_error(smt_process_status,NormalizedStatus)`

`Diagnostics` is a partial list or a list with an element `Element` which is a variable:

`instantiation_error`

`Diagnostics` is neither a partial list nor a list of character codes:

`type_error(codes,Diagnostics)`

`SourceMap` is a partial list or a list with an element `Element` which is a variable:

`instantiation_error`

`SourceMap` is neither a variable nor a list:

`type_error(list,SourceMap)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.299.4 `smt_problem_protocol`

Protocol for ordered, ground SMT-LIB 2 statement providers.

Availability:

`logtalk_load(smt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-08

Compilation flags:

`static`

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *smt_statement/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

smt_statement/2

Enumerates, by backtracking, canonical SMT-LIB 2 statements in source order and their origins.

Compilation flags:
`static`

Template:
`smt_statement(Statement,Origin)`
Mode and number of proofs:
`smt_statement(-compound,-term)` - `zero_or_more`

Remarks:

- Origin: Either `source(File, Start, End)` or `unknown`.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.299.5 smt_process

Isolated operating-system shell transport for the Z3, cvc5, Bitwuzla, and Yices 2 CLIs.

Availability:
`logtalk_load(smt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static, context_switching_calls`

Uses:
list
os
process
reader
type
user
utf_8

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *quote/3*
  - *run/7*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

quote/3

Quotes an atom as one POSIX shell or Windows PowerShell literal, without interpolation.

Compilation flags:
`static`

Template:
`quote(Platform,Argument,Quoted)`
Mode and number of proofs:

`quote(+atom,+atom,-atom)` - `one_or_error`

Exceptions:

`Platform` is a variable:
`instantiation_error`
`Platform` is neither a variable nor an atom:
`type_error(atom,Platform)`
`Platform` is an atom but neither `posix` nor `windows`:
`domain_error(smt_shell_platform,Platform)`
`Argument` is a variable:
`instantiation_error`
`Argument` is neither a variable nor an atom:
`type_error(atom,Argument)`
`Argument` is an atom but contains a null character:
`domain_error(smt_shell_argument,Argument)`

---

run/7

Runs the CLI using an owned directory and separate UTF-8 output files, returning only after cleanup.

Compilation flags:

`static`

Template:

`run(Executable,Arguments,Source,Codes,Diagnostics,RawStatus,NormalizedStatus)`

Mode and number of proofs:

`run(+atom,+list(atom),+compound,-list(code),-list(code),-term,-compound)` - `one_or_error`

Exceptions:

`Executable` is a variable:
`instantiation_error`
`Executable` is neither a variable nor an atom:
`type_error(atom,Executable)`
`Arguments` is a partial list or a list with an element `Argument` which is a variable:
`instantiation_error`
`Arguments` is neither a variable nor a list:
`type_error(list(atom),Arguments)`
An element `Argument` of the list `Arguments` is neither a variable nor an atom:
`type_error(atom,Argument)`
`Executable` or an element `Argument` of the list `Arguments` contains a null character:
`domain_error(smt_shell_argument,Argument)`
`Source` is not a ground term:
`instantiation_error`
`Source` is neither `file(Path)` nor `codes(Codes)`:
`domain_error(smt_process_source,Source)`
A launcher dependency does not exist:

```
existence_error(command,Command)
```
A source or capture file does not exist:
```
existence_error(source_sink,Path)
```
A source or capture file cannot be opened:
```
permission_error(open,source_sink,Path)
```
An owned directory cannot be found during cleanup:
```
existence_error(directory,Directory)
```
An owned file or directory cannot be removed:
```
permission_error(delete,source_sink,Path)
```
A character sequence is not valid UTF-8:
```
representation_error(utf_8)
```
A launcher operation fails:
```
smt_transport(Stage,Status,Details)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.299.6 `smt_program`

Canonical SMT-LIB 2 statement validation, source generation, and response-producing statement classification.

Availability:
```
logtalk_load(smt(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-14

Compilation flags:
```
static, context_switching_calls
```

Uses:
list
type

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *serialize/2*
  - *serialize/4*
  - *response_producing/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

serialize/2

Serializes ground canonical SMT-LIB 2 statements to character codes, auto-declaring `:produce-models` and `:produce-unsat-cores` options when required and not already set by the caller.

Compilation flags:
`static`

Template:
`serialize(Statements,Codes)`

Mode and number of proofs:
`serialize(+list,-list(code))` - `one_or_error`

Exceptions:
`Statements` is a partial list or a list with an element `Statement` which is a variable:
`instantiation_error`
`Statements` is neither a partial list nor a list:
`type_error(list,Statements)`
An element `Statement` of the list `Statements` is not a valid statement:
`domain_error(smt_statement,Statement)`

---

serialize/4

Serializes statements with origins and returns `line(Start, End, Origin)` source-map entries; auto-declared option statements use the `unknown` origin.

Compilation flags:
`static`

Template:
`serialize(Statements,Origins,Codes,SourceMap)`

Mode and number of proofs:
`serialize(+list,+list,-list(code),-list(compound))` - `one_or_error`

Exceptions:

`Statements` is a partial list or a list with an element `Statement` which is a variable:

```
instantiation_error
```

`Statements` is neither a partial list nor a list:

```
type_error(list,Statements)
```

An element `Statement` of the list `Statements` is not a valid statement:

```
domain_error(smt_statement,Statement)
```

`Origins` is a partial list or a list with an element `Origin` which is a variable:

```
instantiation_error
```

`Origins` is neither a partial list nor a list:

```
type_error(list,Origins)
```

An element of the list `Origins` is neither `unknown` nor a valid `source(File, Start, End)` term, or its length differs from `Statements`:

```
domain_error(smt_origins,Origins)
```

`response_producing/1`

Succeeds when `Statement` is a response-producing command, including `get-objectives`.

Compilation flags:

```
static
```

Template:

```
response_producing(Statement)
```

Mode and number of proofs:

`response_producing(+compound)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.299.7 smt_protocol

Protocol for process-backed Z3, cvc5, Bitwuzla, and Yices 2 CLI solving, SMT-LIB 2 source generation, and collected result inspection.

Availability:
`logtalk_load(smt(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *solve/2*
  - *solve/3*
  - *program/1*
  - *program/2*
  - *print_program/1*
  - *status/2*
  - *model_binding/2*
  - *model_value/3*
  - *unsat_core_name/2*
  - *result_property/2*
  - *print_result/1*
  - *print_results/1*
  - *capability/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

solve/2

Solves a source with default options using the external SMT solver CLI and returns a collected result per `check-sat` command, in script order, after cleanup. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`.

Compilation flags:

`static`

Template:

`solve(Source,Results)`

Mode and number of proofs:

`solve(+compound,-list(compound))` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `smt_problem_protocol`:

`domain_error(smt_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(smt_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(smt_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

A launcher argument contains a null character:

`domain_error(smt_shell_argument,Argument)`

The selected solver executable or a shell dependency does not exist:

`existence_error(command,Command)`

An owned directory cannot be found during cleanup:

`existence_error(directory,Directory)`

An owned file or directory cannot be removed:

`permission_error(delete,source_sink,Path)`

A character sequence is not valid UTF-8:

`representation_error(utf_8)`

A launcher operation fails:

`smt_transport(Stage,Status,Details)`

Solver output is invalid or inconsistent with its exit status:

`smt_solver_error(Formal,Status,Diagnostics,SourceMap)`

`solve/3`

Solves a source using `solver/1`, `executable/1`, and `time_limit/1` options. Repeated options use the first occurrence. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`.

Compilation flags:

`static`

Template:

`solve(Source,Results,Options)`

Mode and number of proofs:

`solve(+compound,-list(compound),+list(compound))` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `smt_problem_protocol`:

`domain_error(smt_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(smt_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(smt_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

A launcher argument contains a null character:

```
domain_error(smt_shell_argument,Argument)
```

The selected solver executable or a shell dependency does not exist:

```
existence_error(command,Command)
```

An owned directory cannot be found during cleanup:

```
existence_error(directory,Directory)
```

An owned file or directory cannot be removed:

```
permission_error(delete,source_sink,Path)
```

A character sequence is not valid UTF-8:

```
representation_error(utf_8)
```

A launcher operation fails:

```
smt_transport(Stage,Status,Details)
```

Solver output is invalid or inconsistent with its exit status:

```
smt_solver_error(Formal,Status,Diagnostics,SourceMap)
```

`program/1`

Generates and prints SMT-LIB 2 source to the current output, equivalent to `program/2` followed by `print_program/1`, without invoking the solver.

Compilation flags:

```
static
```

Template:

```
program(Source)
```

Mode and number of proofs:

```
program(+compound) - one_or_error
```

Exceptions:

`Source` is not a ground term:

```
instantiation_error
```

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `smt_problem_protocol`:

```
domain_error(smt_source,Source)
```

`Statements` is ground but not a list:

```
type_error(list,Statements)
```

A statement enumerated by the provider object is not a valid statement:

```
domain_error(smt_statement,Statement)
```

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

```
domain_error(smt_origins,Origins)
```

`Codes` is ground but not a list of character codes:

```
type_error(codes,Codes)
```

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

The current output stream does not allow writing:

`permission_error(output,stream,Stream)`

`program/2`

Returns SMT-LIB 2 source codes from `object(Object)`, `terms(Statements)`, `codes(Codes)`, or `file(Path)` without invoking the solver.

Compilation flags:

`static`

Template:

`program(Source,Codes)`

Mode and number of proofs:

`program(+compound,-list(integer))` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `smt_problem_protocol`:

`domain_error(smt_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(smt_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(smt_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:
`permission_error(open,source_sink,Path)`

`print_program/1`

Prints SMT-LIB 2 character codes to the current output without adding a newline or invoking the solver.

Compilation flags:
`static`

Template:
`print_program(Codes)`
Mode and number of proofs:
`print_program(+list(integer))` - `one_or_error`

Exceptions:
`Codes` is not a ground term:
`instantiation_error`
`Codes` is ground but not a list of character codes:
`type_error(codes,Codes)`
An element `Code` of the list `Codes` is ground but not an integer:
`type_error(integer,Code)`
An element `Code` of the list `Codes` is an integer but not a valid character code:
`domain_error(character_code,Code)`
The current output stream does not allow writing:
`permission_error(output,stream,Stream)`

`status/2`

Returns the collected status (`satisfiable`, `unsatisfiable`, or `unknown`) of a single `check-sat` result.

Compilation flags:
`static`

Template:
`status(Result,Status)`
Mode and number of proofs:
`status(+compound,-atom)` - `one_or_error`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected SMT solver result:
`domain_error(smt_result,Result)`

---

model_binding/2

Enumerates, by backtracking, the collected model bindings (`Symbol-Value` pairs) without starting a process. Fails when the result has no model.

Compilation flags:

```
static
```

Template:

```
model_binding(Result,Binding)
```

Mode and number of proofs:

```
model_binding(+compound,-pair) - zero_or_more
```

Exceptions:

`Result` is not a ground term:

```
instantiation_error
```

`Result` is not a valid collected SMT solver result:

```
domain_error(smt_result,Result)
```

---

model_value/3

Looks up the collected model value bound to `Symbol`. Fails when the result has no model or no binding for `Symbol`.

Compilation flags:

```
static
```

Template:

```
model_value(Result,Symbol,Value)
```

Mode and number of proofs:

```
model_value(+compound,+atom,-term) - zero_or_one
```

Exceptions:

`Result` or `Symbol` is a variable:

```
instantiation_error
```

`Symbol` is neither a variable nor an atom:

```
type_error(atom,Symbol)
```

`Result` is not a valid collected SMT solver result:

```
domain_error(smt_result,Result)
```

---

unsat_core_name/2

Enumerates, by backtracking, the named assertions in the collected unsatisfiable core without starting a process. Fails when the result has no core.

Compilation flags:
`static`

Template:
`unsat_core_name(Result,Name)`
Mode and number of proofs:
`unsat_core_name(+compound,-atom)` - `zero_or_more`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected SMT solver result:
`domain_error(smt_result,Result)`

---

result_property/2

Enumerates, by backtracking, `status(Status)` and the stored result properties.

Compilation flags:
`static`

Template:
`result_property(Result,Property)`
Mode and number of proofs:
`result_property(+compound,?compound)` - `zero_or_more`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected SMT solver result:
`domain_error(smt_result,Result)`

---

print_result/1

Prints a single collected result (status and, when present, model bindings and unsat core names) to the current output, without invoking the solver.

Compilation flags:
`static`

Template:

```
print_result(Result)
```

Mode and number of proofs:

```
print_result(+compound) - one_or_error
```

Exceptions:

`Result` is not a ground term:

```
instantiation_error
```

`Result` is not a valid collected SMT solver result:

```
domain_error(smt_result,Result)
```

The current output stream does not allow writing:

```
permission_error(output,stream,Stream)
```

---

print_results/1

Prints each collected result in `Results`, in order, separating consecutive results with a blank line.

Compilation flags:

```
static
```

Template:

```
print_results(Results)
```

Mode and number of proofs:

```
print_results(+list(compound)) - one_or_error
```

Exceptions:

`Results` is not a ground term:

```
instantiation_error
```

`Results` is ground but not a list:

```
type_error(list,Results)
```

An element of the list `Results` is not a valid collected SMT solver result:

```
domain_error(smt_result,Result)
```

The current output stream does not allow writing:

```
permission_error(output,stream,Stream)
```

---

capability/2

Enumerates, by backtracking, advisory `Solver`-`Feature` capability pairs. Purely informational: the library does not enforce these pairs and a solver call may still succeed, fail, or report `unknown` for a feature outside this table.

Compilation flags:

```
static
```

Template:

```
capability(Solver,Feature)
```

Mode and number of proofs:

`capability(?atom,?atom)` - `zero_or_more`

Remarks:

- Feature: One of `quantifiers`, `strings`, `linear_arithmetic`, `non_linear_arithmetic`, `unsat_cores`, `incremental`, `bit_vectors`, `arrays`, `floating_point`, or `algebraic_datatypes`.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.300 smtp

object

1.300.1 `smtp`

Portable SMTP/ESMTP client.

Availability:
`logtalk_load(smtp(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-11

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *smtp_protocol*
Imports:
`public` *options*
Uses:
base64
list
reader
smtp_process_transport
socket
type
user

utf_8

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *connect/4* *connection_alive/1* *default_option/1* *default_options/1* *disconnect/1* *option/2* *option/3* *send/4* *send/5* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.300.2 `smtp_process_transport`

Process-backed TLS transport support for the SMTP client.

Availability:
`logtalk_load(smtp(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-11-12

Compilation flags:
`static, context_switching_calls`

Uses:
list
os
process

reader
user

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *open/8*
  - *streams/3*
  - *failure_details/3*
  - *close/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

open/8

Opens an implicit TLS or STARTTLS SMTP connection using OpenSSL.

Compilation flags:
`static`

Template:
`open(Host,Port,Security,Helo,Executable,ServerName,Options,Connection)`
Mode and number of proofs:
`open(+atom,+integer,+atom,+atom,+atom,+atom,+list,--compound)` - `one_or_error`

---

streams/3

Returns the input and output streams for an open process connection.

Compilation flags:
`static`

Template:
`streams(Connection,Input,Output)`
Mode and number of proofs:
`streams(+compound,--stream,--stream)` - `one_or_error`

Exceptions:

Either `Input` or `Output` streams do not exist:
`domain_error(smtp_process_connection,Connection)`

`failure_details/3`

Returns the process exit status and standard error output after an OpenSSL connection failure.

Compilation flags:
`static`

Template:
`failure_details(Connection,Status,Diagnostic)`
Mode and number of proofs:
`failure_details(+compound,--term,--atom)` - `one`

`close/1`

Closes the streams and terminates the OpenSSL process.

Compilation flags:
`static`

Template:
`close(Connection)`
Mode and number of proofs:
`close(+compound)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.300.3 smtp_protocol

Protocol for portable SMTP clients.

Availability:
`logtalk_load(smtp(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *send/5*
  - *connect/4*
  - *disconnect/1*
  - *connection_alive/1*
  - *send/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

send/5

Opens an SMTP session, sends a message, and closes the session. Unicode `Subject` and `Comments` header values are serialized as RFC 2047 UTF-8 Base64 encoded words; all other header values must be ASCII.

Compilation flags:
`static`

Template:
`send(Host,Port,Message,Result,Options)`
Mode and number of proofs:
`send(+atom,+integer,+compound,--compound,+list)` - `one_or_error`

Exceptions:

`Host` or `Port` is a variable:

```
instantiation_error
```

`Host` is neither a variable nor an atom:

```
type_error(atom,Host)
```

`Port` is neither a variable nor an integer:

```
type_error(integer,Port)
```

`Port` is an integer but not positive:

```
domain_error(positive_integer,Port)
```

`Message` is not an SMTP message term:

```
domain_error(smtp_message,Message)
```

The envelope sender or a recipient is a variable:

```
instantiation_error
```

The envelope sender or a recipient is neither a variable nor an atom:

```
type_error(atom,Mailbox)
```

The envelope sender or a recipient is not a valid mailbox:

```
domain_error(smtp_mailbox,Mailbox)
```

`Recipients` is not a non-empty list of mailboxes or a mailbox atom:

```
domain_error(smtp_recipients,Recipients)
```

`Headers` is not a list:

```
type_error(list,Headers)
```

A header name or value is a variable:

```
instantiation_error
```

A header name or value is neither a variable nor an atom:

```
type_error(atom,HeaderComponent)
```

`Header` is not a valid header pair:

```
domain_error(smtp_header,Header)
```

`Body` is not a supported text representation:

```
domain_error(smtp_body,Body)
```

A caller-supplied MIME header conflicts with UTF-8 Base64 serialization or duplicates another MIME header:

```
domain_error(smtp_mime_header,Header)
```

`Options` is a variable:

```
instantiation_error
```

`Options` is neither a variable nor a list:

```
type_error(list,Options)
```

An element `Option` of the list `Options` is a variable:

```
instantiation_error
```

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

The SMTP connection cannot be opened:

```
smtp_error(connection_failed)
```

OpenSSL terminates the secure connection while an SMTP response is expected:

```
smtp_error(secure_connection_failed(Status,Diagnostic))
```

The SMTP server closes the connection while a response is expected at `Stage`:

```
smtp_error(connection_closed(Stage))
```

The SMTP server rejects the session greeting:

```
smtp_error(greeting_failed(Code,Lines))
```

The SMTP server returns an invalid or unexpected response:

`smtp_error(protocol_error(Response))`

Authentication is requested over an insecure connection without explicit permission:

`permission_error(authenticate,insecure_smtp_connection,plain)`

The SMTP server does not advertise a supported authentication mechanism:

`smtp_error(authentication_not_supported)`

The SMTP server rejects authentication:

`smtp_error(auth_failed(Code,Lines))`

The SMTP server rejects the envelope sender:

`smtp_error(sender_rejected(Code,Lines))`

The SMTP server rejects the DATA command or the submitted message:

`smtp_error(send_failed(Code,Lines))`

---

connect/4

Opens an SMTP session for sending one or more messages.

Compilation flags:

`static`

Template:

`connect(Host,Port,Connection,Options)`

Mode and number of proofs:

`connect(+atom,+integer,--compound,+list)` - `one_or_error`

Exceptions:

`Host` or `Port` is a variable:

`instantiation_error`

`Host` is neither a variable nor an atom:

`type_error(atom,Host)`

`Port` is neither a variable nor an integer:

`type_error(integer,Port)`

`Port` is an integer but not positive:

`domain_error(positive_integer,Port)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Option` is valid but is not a connection option:

`domain_error(smtp_connection_option,Option)`

The SMTP connection cannot be opened:

`smtp_error(connection_failed)`

OpenSSL terminates the secure connection while an SMTP response is expected:
`smtp_error(secure_connection_failed(Status,Diagnostic))`
The SMTP server closes the connection while a response is expected at `Stage`:
`smtp_error(connection_closed(Stage))`
The SMTP server rejects the session greeting:
`smtp_error(greeting_failed(Code,Lines))`
The SMTP server returns an invalid or unexpected response:
`smtp_error(protocol_error(Response))`
Authentication is requested over an insecure connection without explicit permission:
`permission_error(authenticate,insecure_smtp_connection,plain)`
The SMTP server does not advertise a supported authentication mechanism:
`smtp_error(authentication_not_supported)`
The SMTP server rejects authentication:
`smtp_error(auth_failed(Code,Lines))`

---

disconnect/1

Gracefully closes an SMTP session.

Compilation flags:
`static`

Template:
`disconnect(Connection)`
Mode and number of proofs:
`disconnect(+compound)` - `one_or_error`

Exceptions:
`Connection` is not an open SMTP connection handle:
`domain_error(smtp_connection,Connection)`

---

connection_alive/1

Checks if the local streams for an SMTP connection are open.

Compilation flags:
`static`

Template:
`connection_alive(Connection)`
Mode and number of proofs:
`connection_alive(+compound)` - `zero_or_one`

---

send/4

Sends a message over an open SMTP session. Unicode `Subject` and `Comments` header values are serialized as RFC 2047 UTF-8 Base64 encoded words; all other header values must be ASCII.

Compilation flags:

`static`

Template:

`send(Connection,Message,Result,Options)`

Mode and number of proofs:

`send(+compound,+compound,--compound,+list)` - `one_or_error`

Exceptions:

`Connection` is not an open SMTP connection handle:

`domain_error(smtp_connection,Connection)`

`Message` is not an SMTP message term:

`domain_error(smtp_message,Message)`

The envelope sender or a recipient is a variable:

`instantiation_error`

The envelope sender or a recipient is neither a variable nor an atom:

`type_error(atom,Mailbox)`

The envelope sender or a recipient is not a valid mailbox:

`domain_error(smtp_mailbox,Mailbox)`

`Recipients` is not a non-empty list of mailboxes or a mailbox atom:

`domain_error(smtp_recipients,Recipients)`

`Headers` is not a list:

`type_error(list,Headers)`

A header name or value is a variable:

`instantiation_error`

A header name or value is neither a variable nor an atom:

`type_error(atom,HeaderComponent)`

`Header` is not a valid header pair:

`domain_error(smtp_header,Header)`

`Body` is not a supported text representation:

`domain_error(smtp_body,Body)`

A caller-supplied MIME header conflicts with UTF-8 Base64 serialization or duplicates another MIME header:

`domain_error(smtp_mime_header,Header)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

`Option` is valid but is not a transaction option:

`domain_error(smtp_transaction_option,Option)`

The SMTP server closes the connection while a response is expected at `Stage`:
`smtp_error(connection_closed(Stage))`
The SMTP server returns an invalid or unexpected response:
`smtp_error(protocol_error(Response))`
The SMTP server rejects the envelope sender:
`smtp_error(sender_rejected(Code,Lines))`
The SMTP server rejects the DATA command or the submitted message:
`smtp_error(send_failed(Code,Lines))`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.301 snowflakeid

object

1.301.1 `snowflakeid`

Snowflake ID generator using the Twitter-style profile and atom representation.

Availability:
`logtalk_load(snowflakeid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *snowflakeid_twitter(atom)*

Remarks:
(none)

Inherited public predicates:
generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node), *snowflakeid_twitter*, *snowflakeid_sonyflake*, *snowflakeid_instagram*, *cuid2*, *ids*, *ksuid*, *nanoid*, *sqids*, *typeid*, *ulid*, *uuid*

object

1.301.2 `snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node)`

- `Representation` - Snowflake ID representation. Possible values are `integer`, `atom`, `chars`, and `codes`.
- `EpochMilliseconds` - Custom epoch in Unix milliseconds.
- `TimeUnitMilliseconds` - Timestamp unit in milliseconds.
- `TimestampBits` - Number of timestamp bits.
- `NodeBits` - Number of node bits.
- `SequenceBits` - Number of sequence bits.
- `Node` - Node identifier value.

Snowflake ID generic generator.

Availability:
`logtalk_load(snowflakeid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *snowflakeid_protocol*
Uses:
iso8601
os

Remarks:
(none)

Inherited public predicates:
generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *last_time_sequence_/2*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

last_time_sequence_/2

Last time and sequence.

Compilation flags:
`dynamic`

Template:
`last_time_sequence_(Time,Sequence)`
Mode and number of proofs:
`last_time_sequence_(?integer,?integer)` - `zero_or_one`

Operators

(none)

See also

snowflakeid, *snowflakeid_twitter*, *snowflakeid_sonyflake*, *snowflakeid_instagram*, *cuid2(Representation,Size,Alphabet)*, *ids(Representation,Bytes)*, *ksuid(Representation,Alphabet)*, *nanoid(Representation,Size,Alphabet)*, *sqids(Representation)*, *typeid(Representation)*, *ulid(Representation)*, *uuid(Representation)*

object

1.301.3 `snowflakeid_instagram`

Instagram-style Snowflake profile using atom representation.

Availability:

```
logtalk_load(snowflakeid(loader))
```

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:

```
static, context_switching_calls
```

Extends:

`public` *snowflakeid_instagram(atom)*

Remarks:

(none)

Inherited public predicates:

generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

snowflakeid_instagram(Representation), *snowflakeid_twitter*, *snowflakeid_sonyflake*, *cuid2*, *ids*, *ksuid*, *nanoid*, *sqids*, *typeid*, *ulid*, *uuid*

object

1.301.4 `snowflakeid_instagram(Representation)`

- `Representation` - Snowflake ID representation. Possible values are `integer`, `atom`, `chars`, and `codes`.

Instagram-style Snowflake profile.

Availability:
`logtalk_load(snowflakeid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *snowflakeid(Representation,1314220021721,1,41,13,10,1)*

Remarks:
(none)

Inherited public predicates:
generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node), *snowflakeid_twitter*, *snowflakeid_sonyflake*, *snowflakeid_instagram*, *cuid2*, *ids*, *ksuid*, *nanoid*, *sqids*, *typeid*, *ulid*, *uuid*

protocol

1.301.5 `snowflakeid_protocol`

Snowflake ID generator protocol.

Availability:
`logtalk_load(snowflakeid(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-26

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *generate/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

generate/1

Returns a Snowflake ID.

Compilation flags:
`static`

Template:
`generate(ID)`
Mode and number of proofs:
`generate(--ground)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.301.6 snowflakeid_sonyflake

Sonyflake-style Snowflake profile using atom representation.

Availability:
`logtalk_load(snowflakeid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *snowflakeid_sonyflake(atom)*

Remarks:
(none)

Inherited public predicates:
generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

snowflakeid_sonyflake(Representation), *snowflakeid_twitter*, *snowflakeid_instagram*, *cuid2*, *ids*, *ksuid*, *nanoid*, *sqids*, *typeid*, *ulid*, *uuid*

object

1.301.7 `snowflakeid_sonyflake(Representation)`

- `Representation` - Snowflake ID representation. Possible values are `integer`, `atom`, `chars`, and `codes`.

Sonyflake-style Snowflake profile.

Availability:
`logtalk_load(snowflakeid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *snowflakeid(Representation,1409529600000,10,39,16,8,1)*

Remarks:
(none)

Inherited public predicates:

generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node), *snowflakeid_twitter*, *snowflakeid_sonyflake*, *snowflakeid_instagram*, *cuid2*, *ids*, *ksuid*, *nanoid*, *sqids*, *typeid*, *ulid*, *uuid*

object

1.301.8 `snowflakeid_twitter`

Twitter-style Snowflake profile using atom representation.

Availability:
`logtalk_load(snowflakeid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-04-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *snowflakeid_twitter(atom)*

Remarks:
(none)

Inherited public predicates:

generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

snowflakeid_twitter(Representation), *snowflakeid_sonyflake*, *snowflakeid_instagram*, *cuid2*, *ids*, *ksuid*, *nanoid*, *sqids*, *typeid*, *ulid*, *uuid*

object

1.301.9 `snowflakeid_twitter(Representation)`

- `Representation` - Snowflake ID representation. Possible values are `integer`, `atom`, `chars`, and `codes`.

Twitter-style Snowflake profile.

Availability:

`logtalk_load(snowflakeid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-25

Compilation flags:

`static, context_switching_calls`

Extends:

`public` *snowflakeid(Representation,1288834974657,1,41,10,12,1)*

Remarks:

(none)

Inherited public predicates:

generate/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node), *snowflakeid_twitter*, *snowflakeid_sonyflake*, *snowflakeid_instagram*, *cuid2*, *ids*, *ksuid*, *nanoid*, *sqids*, *typeid*, *ulid*, *uuid*

1.302 sockets

object

1.302.1 `socket`

Portable abstraction over TCP sockets. Provides a high-level API for client and server socket operations that works with selected backend Prolog systems.

Availability:

```
logtalk_load(sockets(loader))
```

Author: Paulo Moura
Version: 0:14:1
Date: 2026-11-18

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *options*

Uses:

list

Remarks:

- Supported backends: ECLiPSe, GNU Prolog, SICStus Prolog, Shumway Prolog, SWI-Prolog, Trealla Prolog, and XVM.
- Design rationale: Some backends (notably SICStus Prolog) do not provide low-level socket creation predicates that can be separated from binding or connecting. This library therefore provides a higher-level API with `client_open/5` and `server_open/3` that abstracts over these differences.
- Stream handling: Predicates `client_open/5` and `server_accept/5` return separate input and output streams opened in binary mode. For backends where the same stream is used for bidirectional communication, the same stream handle is returned in both arguments. Use standard stream predicates (`put_byte/2`, `get_byte/2`, `read/2`, `write/2`, etc.) to communicate.
- Options: Currently options are only defined for the `server_open/3` predicate. This is expected to change in future versions for the other predicates that have an options argument.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *client_open/5*
  - *client_open/4*
  - *server_open/4*
  - *server_open/3*
  - *server_open/2*
  - *server_accept/5*
  - *server_accept/4*
  - *server_close/1*
  - *close/2*
  - *current_host/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

client_open/5

Opens a client connection to the specified host and port using the given options. Returns separate input and output streams for bidirectional communication. The streams are opened by default in binary mode.

Compilation flags:

`static`

Template:

```
client_open(Host,Port,InputStream,OutputStream,Options)
```

Mode and number of proofs:

```
client_open(+atom,+integer,--stream,--stream,+list) - one_or_error
```

Exceptions:

Connection refused or host not found:

```
socket_error(Error)
```

Remarks:

- Option `type(binary)`: Open the streams in binary mode. This is the default.
- Option `type(text)`: Open the streams in text mode.

---

client_open/4

Opens a client connection to the specified host and port using default options. Returns separate input and output streams for bidirectional communication. The streams are opened in binary mode.

Compilation flags:

```
static
```

Template:

```
client_open(Host,Port,InputStream,OutputStream)
```

Mode and number of proofs:

```
client_open(+atom,+integer,--stream,--stream) - one_or_error
```

Exceptions:

Connection refused or host not found:

```
socket_error(Error)
```

---

server_open/4

Opens a server socket bound to the specified host and port using the given options. If `Port` is a variable, binds to an available port and unifies `Port` with the port number. Returns a `ServerSocket` handle to use with `server_accept/4`. The default backlog (queue length) for pending connections is 5. Use the option `backlog(N)` to override. This option is not supported and thus ignored by the SICStus Prolog and Trealla Prolog backends.

Compilation flags:

```
static
```

Template:

```
server_open(Host,Port,ServerSocket,Options)
```

Mode and number of proofs:

```
server_open(+atom,?integer,--compound,+list) - one_or_error
```

Exceptions:

Port already in use:
`socket_error(Error)`

---

server_open/3

Opens a server socket bound to the specified port using the given options. If `Port` is a variable, binds to an available port and unifies `Port` with the port number. Returns a `ServerSocket` handle to use with `server_accept/4`. The default backlog (queue length) for pending connections is 5. Use the option `backlog(N)` to override. This option is not supported and thus ignored by the SICStus Prolog and Trealla Prolog backends.

Compilation flags:
`static`

Template:
`server_open(Port,ServerSocket,Options)`
Mode and number of proofs:
`server_open(?integer,--compound,+list)` - `one_or_error`

Exceptions:
Port already in use:
`socket_error(Error)`

---

server_open/2

Opens a server socket bound to the specified port using default options. If `Port` is a variable, binds to an available port and unifies `Port` with the port number. Returns a `ServerSocket` handle to use with `server_accept/4`. The default backlog (queue length) for pending connections is 5.

Compilation flags:
`static`

Template:
`server_open(Port,ServerSocket)`
Mode and number of proofs:
`server_open(?integer,--compound)` - `one_or_error`

Exceptions:
Port already in use:
`socket_error(Error)`

---

server_accept/5

Accepts an incoming connection on the server socket, blocking until a client connects, using the given options. Returns separate input and output streams for bidirectional communication and client information as `client(Host, Port)` or `client(Address)` depending on backend. The streams are opened by default in binary mode.

Compilation flags:

`static`

Template:

`server_accept(ServerSocket,InputStream,OutputStream,ClientInfo,Options)`

Mode and number of proofs:

`server_accept(+compound,--stream,--stream,--compound,+list)` - `one_or_error`

Exceptions:

Invalid server socket:

`socket_error(Error)`

Remarks:

- Option `type(binary)`: Open the streams in binary mode. This is the default.
- Option `type(text)`: Open the streams in text mode.

---

server_accept/4

Accepts an incoming connection on the server socket, blocking until a client connects, using default options. Returns separate input and output streams for bidirectional communication and client information as `client(Host, Port)` or `client(Address)` depending on backend. The streams are opened in binary mode.

Compilation flags:

`static`

Template:

`server_accept(ServerSocket,InputStream,OutputStream,ClientInfo)`

Mode and number of proofs:

`server_accept(+compound,--stream,--stream,--compound)` - `one_or_error`

Exceptions:

Invalid server socket:

`socket_error(Error)`

---

server_close/1

Closes a server socket.

Compilation flags:
`static`

Template:
`server_close(ServerSocket)`
Mode and number of proofs:
`server_close(+compound)` - `one_or_error`

---

close/2

Closes a client or accepted connection by closing both the input and output streams. If the same stream is used for both, it is closed only once.

Compilation flags:
`static`

Template:
`close(InputStream,OutputStream)`
Mode and number of proofs:
`close(+stream,+stream)` - `one_or_error`

Exceptions:
Error when trying to close the socket streams:
`socket_error(Error)`

---

current_host/1

Returns the host name of the current machine.

Compilation flags:
`static`

Template:
`current_host(Host)`
Mode and number of proofs:
`current_host(-atom)` - `one_or_error`

Exceptions:
Error when trying to get the current host name:
`socket_error(Error)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.303 souffle

object

1.303.1 `souffle`

Process-backed Souffle CLI solving, Datalog source generation, and collected result inspection.

Availability:
`logtalk_load(souffle(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-10

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *souffle_protocol*
Imports:
`public` *options*
Uses:
list
os
reader
souffle_output
souffle_process
souffle_program
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *print_program/1* *print_result/1* *program/1* *program/2* *relation/4* *result_property/2* *size/3* *solve/2* *solve/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.303.2 `souffle_expansion`

Opt-in ordinary-clause and directive expansion for pure Datalog problem objects.

Availability:

```
logtalk_load(souffle(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-15

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *expanding*

Uses:

list
souffle_program
type

Remarks:

(none)

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.303.3 `souffle_output`

Decoding of captured stdout (`.printsize` lines) and per-relation tab-separated CSV file content into a collected result.

Availability:
`logtalk_load(souffle(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-15

Compilation flags:
`static, context_switching_calls`

Uses:
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*

  - *decode/8*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

decode/8

Decodes captured stdout, stderr, per-relation CSV file content, and process metadata into a collected `souffle_result(Relations, Sizes, Properties)` term.

Compilation flags:

`static`

Template:

`decode(Codes,Diagnostics,RelationCodes,RawStatus,NormalizedStatus,Outputs,Printsizes,Result)`

Mode and number of proofs:

`decode(+list(code),+list(code),+list(compound),+term,+compound,+list(pair),+list(atom),-compound)` - `one_or_error`

Exceptions:

`Codes` is a partial list or a list with an element `Element` which is a variable:

`instantiation_error`

`Codes` is neither a partial list nor a list of character codes:

`type_error(codes,Codes)`

`Diagnostics` is a partial list or a list with an element `Element` which is a variable:

`instantiation_error`

`Diagnostics` is neither a partial list nor a list of character codes:

`type_error(codes,Diagnostics)`

`RelationCodes`, `Outputs`, or `Printsizes` is a partial list or has a variable element:

`instantiation_error`

`RelationCodes`, `Outputs`, or `Printsizes` is neither a partial list nor a list:

`type_error(list,RelationCodes)`

`RawStatus` is not a ground term:

`instantiation_error`

`NormalizedStatus` is not a ground term:

`instantiation_error`

`NormalizedStatus` is not `exit(0)`:

`domain_error(souffle_process_status,NormalizedStatus)`

A requested relation in `Outputs` has no entry, or a `missing` entry, in `RelationCodes`:

`domain_error(souffle_output,Name-missing)`

A CSV row for a requested relation has a different number of columns than declared in `Outputs`:

`domain_error(souffle_output,column_count-(Row-Types))`

A `number` or `unsigned` column value is not a valid integer, or an `unsigned` value is negative:

`domain_error(souffle_output,Type-Value)`

A `float` column value is not a valid number:

`domain_error(souffle_output,float-Value)`

A requested relation in `Printsizes` has no matching `Name\tCount` line in `Codes`:
`domain_error(souffle_output,Name-missing)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.303.4 `souffle_problem_protocol`

Protocol for ordered, ground Datalog statement providers.

Availability:
`logtalk_load(souffle(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-10

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *souffle_statement/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

souffle_statement/2

Enumerates, by backtracking, canonical Datalog statements in source order and their origins.

Compilation flags:
`static`

Template:
`souffle_statement(Statement,Origin)`
Mode and number of proofs:
`souffle_statement(-compound,-term)` - `zero_or_more`

Remarks:

- Origin: Either `source(File, Start, End)` or `unknown`.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.303.5 souffle_process

Isolated operating-system shell transport for the Souffle CLI, including owned facts and output directories.

Availability:
`logtalk_load(souffle(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-10

Compilation flags:
`static, context_switching_calls`

Uses:
list
os
process
reader

type
user
utf_8

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *quote/3*
  - *run/10*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

quote/3

Quotes an atom as one POSIX shell or Windows PowerShell literal, without interpolation.

Compilation flags:
`static`

Template:
`quote(Platform,Argument,Quoted)`
Mode and number of proofs:
`quote(+atom,+atom,-atom)` - `one_or_error`

Exceptions:
`Platform` is a variable:
`instantiation_error`
`Argument` is a variable:
`instantiation_error`
`Platform` is neither a variable nor an atom:
`type_error(atom,Platform)`
`Argument` is neither a variable nor an atom:
`type_error(atom,Argument)`
`Platform` is an atom but neither `posix` nor `windows`:
`domain_error(souffle_shell_platform,Platform)`
`Argument` is an atom but contains a null character:
`domain_error(souffle_shell_argument,Argument)`

run/10

Runs the CLI using an owned directory and separate UTF-8 capture files, materializing EDB facts and reading back requested `.output` relation files before cleanup, returning only after cleanup.

Compilation flags:

`static`

Template:

`run(Executable,Arguments,ProgramSource,FactsSpec,OutputRelations,Codes,Diagnostics,RelationCodes,RawStatus,NormalizedStatus)`

Mode and number of proofs:

`run(+atom,+list(atom),+compound,+compound,+list(atom),-list(integer),-list(integer),-list(compound),-term,-compound)` - `one_or_error`

Exceptions:

`Executable`, `Arguments`, `ProgramSource`, `FactsSpec`, or `OutputRelations` is a variable:

`instantiation_error`

An element of the list `Arguments` or `OutputRelations` is a variable:

`instantiation_error`

`Executable` is neither a variable nor an atom:

`type_error(atom,Executable)`

`Arguments` is neither a variable nor a list:

`type_error(list,Arguments)`

An element of the list `Arguments` is neither a variable nor an atom:

`type_error(atom,Argument)`

An element of the list `OutputRelations` is neither a variable nor an atom:

`type_error(atom,Name)`

`ProgramSource` is neither `file(Path)` nor `codes(Codes)`:

`domain_error(souffle_process_source,ProgramSource)`

`FactsSpec` is neither `none`, `external(Path)`, nor `materialize(Pairs)`:

`domain_error(souffle_facts_spec,FactsSpec)`

A materialized facts row has a symbol column value containing a tab or newline character, or a value that is not an integer, float, or atom:

`domain_error(souffle_facts_value,Value)`

An argument contains a null character:

`domain_error(souffle_shell_argument,Argument)`

A launcher dependency does not exist:

`existence_error(command,Command)`

A source or capture file does not exist:

`existence_error(source_sink,Path)`

A source or capture file cannot be opened:

`permission_error(open,source_sink,Path)`

An owned directory cannot be found during cleanup:

`existence_error(directory,Directory)`

An owned file or directory cannot be removed:

`permission_error(delete,source_sink,Path)`

A character sequence is not valid UTF-8:

`representation_error(utf_8)`

A launcher operation fails:

```
souffle_transport(Stage,Status,Details)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.303.6 `souffle_program`

Canonical Datalog statement validation and source generation for Souffle 2.5.

Availability:
`logtalk_load(souffle(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-15

Compilation flags:
`static, context_switching_calls`

Uses:
list
type

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *serialize/2*
  - *serialize/4*
  - *valid_identifier/1*
  - *valid_column_type/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

serialize/2

Serializes ground canonical Datalog statements to character codes.

Compilation flags:
`static`

Template:
`serialize(Statements,Codes)`
Mode and number of proofs:
`serialize(+list,-list(integer))` - `one_or_error`

Exceptions:
`Statements` is a partial list or a list with an element `Statement` which is a variable:
`instantiation_error`
`Statements` is neither a partial list nor a list:
`type_error(list,Statements)`
An element `Statement` of the list `Statements` is not a valid statement:
`domain_error(souffle_statement,Statement)`

---

serialize/4

Serializes statements with origins and returns `line(Start, End, Origin)` source-map entries.

Compilation flags:
`static`

Template:
`serialize(Statements,Origins,Codes,SourceMap)`
Mode and number of proofs:
`serialize(+list,+list,-list(code),-list(compound))` - `one_or_error`

Exceptions:
`Statements` is a partial list or a list with an element `Statement` which is a variable:
`instantiation_error`
`Statements` is neither a partial list nor a list:
`type_error(list,Statements)`
An element `Statement` of the list `Statements` is not a valid statement:
`domain_error(souffle_statement,Statement)`
`Origins` is a partial list or a list with an element `Origin` which is a variable:
`instantiation_error`
`Origins` is neither a variable nor a list:
`type_error(list,Origins)`
`Origins` has an element that is neither `unknown` nor a valid `source(File, Start, End)` term, or its length differs from `Statements`:
`domain_error(souffle_origins,Origins)`

---

valid_identifier/1

True if the argument is a supported Datalog relation, column, or type identifier (a lowercase or underscore led name).

Compilation flags:
`static`

Template:
`valid_identifier(Name)`
Mode and number of proofs:
`valid_identifier(@term)` - `zero_or_one`

---

valid_column_type/1

True if the argument is one of the four supported primitive column types.

Compilation flags:
`static`

Template:
`valid_column_type(Type)`
Mode and number of proofs:
`valid_column_type(@term)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.303.7 souffle_protocol

Protocol for process-backed Souffle Datalog solving, source generation, and collected result inspection.

Availability:
`logtalk_load(souffle(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *solve/2*
  - *solve/3*
  - *program/1*
  - *program/2*
  - *print_program/1*
  - *relation/4*
  - *size/3*
  - *result_property/2*
  - *print_result/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

solve/2

Solves a source with default options using the external Souffle CLI and returns a collected result after cleanup. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`. No `.output`/`.printsize` relations are decoded unless requested via the `outputs/1`/`printsizes/1` options of `solve/3`.

Compilation flags:
`static`

Template:

`solve(Source,Result)`

Mode and number of proofs:

`solve(+compound,-compound)` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `souffle_problem_protocol`:

`domain_error(souffle_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(souffle_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(souffle_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

A launcher argument contains a null character:

`domain_error(souffle_shell_argument,Argument)`

The `souffle` executable or a shell dependency does not exist:

`existence_error(command,Command)`

An owned directory cannot be found during cleanup:

`existence_error(directory,Directory)`

An owned file or directory cannot be removed:

`permission_error(delete,source_sink,Path)`

A character sequence is not valid UTF-8:

`representation_error(utf_8)`

A launcher operation fails:

`souffle_transport(Stage,Status,Details)`

Solver output is invalid or inconsistent with its exit status:

`souffle_solver_error(Formal,Status,Diagnostics,SourceMap)`

solve/3

Solves a source using executable, parallelism, warning, compilation, EDB facts, and decoded-relation options. Repeated options use the first occurrence. Valid sources are `object(Object)`, `terms(Statements)`, `codes(Codes)`, and `file(Path)`. The `outputs/1` option lists which `.output` relations to decode and their column types; the `printsizes/1` option lists which `.printsize` relations to decode.

Compilation flags:

`static`

Template:

`solve(Source,Result,Options)`

Mode and number of proofs:

`solve(+compound,-compound,+list(compound))` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `souffle_problem_protocol`:

`domain_error(souffle_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(souffle_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(souffle_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

Both `facts/1` and `facts_data/1` are given with non-default values:

`domain_error(souffle_facts_option,facts(Path)-facts_data(Pairs))`

An element of `facts_data/1` has a symbol column value containing a tab or newline character, or a value that is not an integer, float, or atom:

`domain_error(souffle_facts_value,Value)`

A launcher argument contains a null character:

`domain_error(souffle_shell_argument,Argument)`

The `souffle` executable or a shell dependency does not exist:

`existence_error(command,Command)`

An owned directory cannot be found during cleanup:

`existence_error(directory,Directory)`

An owned file or directory cannot be removed:

`permission_error(delete,source_sink,Path)`

A character sequence is not valid UTF-8:

`representation_error(utf_8)`

A launcher operation fails:

`souffle_transport(Stage,Status,Details)`

Solver output is invalid or inconsistent with its exit status:

`souffle_solver_error(Formal,Status,Diagnostics,SourceMap)`

`program/1`

Generates and prints Datalog source to the current output, equivalent to `program/2` followed by `print_program/1`, without invoking Souffle.

Compilation flags:

`static`

Template:

`program(Source)`

Mode and number of proofs:

`program(+compound)` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `souffle_problem_protocol`:

`domain_error(souffle_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(souffle_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(souffle_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:

`permission_error(open,source_sink,Path)`

The current output stream does not allow writing:

`permission_error(output,stream,Stream)`

---

program/2

Returns Datalog source codes from `object(Object)`, `terms(Statements)`, `codes(Codes)`, or `file(Path)` without invoking Souffle.

Compilation flags:

`static`

Template:

`program(Source,Codes)`

Mode and number of proofs:

`program(+compound,-list(integer))` - `one_or_error`

Exceptions:

`Source` is not a ground term:

`instantiation_error`

`Source` is ground but not a supported source, or is `object(Object)` where `Object` does not implement `souffle_problem_protocol`:

`domain_error(souffle_source,Source)`

`Statements` is ground but not a list:

`type_error(list,Statements)`

A statement enumerated by the provider object is not a valid statement:

`domain_error(souffle_statement,Statement)`

An origin enumerated by the provider object is invalid, or its origins and statements differ in length:

`domain_error(souffle_origins,Origins)`

`Codes` is ground but not a list of character codes:

`type_error(codes,Codes)`

An element `Code` of the list `Codes` is ground but not an integer:

`type_error(integer,Code)`

An element `Code` of the list `Codes` is an integer but not a valid character code:

`domain_error(character_code,Code)`

`Path` is ground but not an atom:

`type_error(atom,Path)`

`Path` does not exist:

`existence_error(source_sink,Path)`

`Path` cannot be opened for reading:
`permission_error(open,source_sink,Path)`

---

print_program/1

Prints Datalog character codes to the current output without adding a newline or invoking Souffle.

Compilation flags:
`static`

Template:
`print_program(Codes)`
Mode and number of proofs:
`print_program(+list(integer))` - `one_or_error`

Exceptions:
`Codes` is not a ground term:
`instantiation_error`
`Codes` is ground but not a list of character codes:
`type_error(codes,Codes)`
An element `Code` of the list `Codes` is ground but not an integer:
`type_error(integer,Code)`
An element `Code` of the list `Codes` is an integer but not a valid character code:
`domain_error(character_code,Code)`
The current output stream does not allow writing:
`permission_error(output,stream,Stream)`

---

relation/4

Enumerates, by backtracking, decoded `.output` relations collected in a result: the relation name, its requested column types, and its rows (each row a list of decoded Prolog values, one per column).

Compilation flags:
`static`

Template:
`relation(Result,Name,Columns,Rows)`
Mode and number of proofs:
`relation(+compound,?atom,?list(atom),?list(list))` - `zero_or_more`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected Souffle result:
`domain_error(souffle_result,Result)`

---

size/3

Enumerates, by backtracking, decoded `.printsize` relation counts collected in a result.

Compilation flags:
`static`

Template:
`size(Result,Name,Count)`
Mode and number of proofs:
`size(+compound,?atom,?integer)` - `zero_or_more`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected Souffle result:
`domain_error(souffle_result,Result)`

---

result_property/2

Enumerates, by backtracking, the stored result properties (`raw_status/1`, `process_status/1`, and `diagnostics/1`).

Compilation flags:
`static`

Template:
`result_property(Result,Property)`
Mode and number of proofs:
`result_property(+compound,?compound)` - `zero_or_more`

Exceptions:
`Result` is not a ground term:
`instantiation_error`
`Result` is not a valid collected Souffle result:
`domain_error(souffle_result,Result)`

---

print_result/1

Prints collected relations and printsize counts to the current output, without invoking Souffle.

Compilation flags:
`static`

Template:
`print_result(Result)`

Mode and number of proofs:

`print_result(+compound)` - `one_or_error`

Exceptions:

`Result` is not a ground term:

`instantiation_error`

`Result` is not a valid collected Souffle result:

`domain_error(souffle_result,Result)`

The current output stream does not allow writing:

`permission_error(output,stream,Stream)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.304 spade_pattern_miner

object

1.304.1 `spade_pattern_miner`

SPADE sequential pattern miner for sequence datasets using vertical occurrence lists.

Availability:

`logtalk_load(spade_pattern_miner(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *sequential_pattern_mining_common*

Uses:

avltree
format
list

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_pattern_miner/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *mine/2* *mine/3* *option/2* *option/3* *pattern_miner_options/2* *print_pattern_miner/1* *valid_option/1* *valid_options/1* *valid_pattern_miner/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pattern_miner_protocol, *sequence_dataset_protocol*, *prefix_span_pattern_miner*, *gsp_pattern_miner*, *clo_span_pattern_miner*

1.305 spectral_clusterer

object

1.305.1 `spectral_clusterer`

Normalized spectral clusterer for continuous datasets using a Gaussian RBF affinity and a Nystroem extension for assigning new instances to clusters.

Availability:
```
logtalk_load(spectral_clusterer(loader))
```

Author: Paulo Moura
Version: 2:1:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *clusterer_common*

Uses:
avltree
format
linear_algebra
list
numberlist
pairs
type

Remarks:
(none)

Inherited public predicates:
check_clusterer/1 *check_option/1* *check_options/1* *cluster/3* *clusterer_options/2* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_clusterer/1* *valid_clusterer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *clusterer_diagnostics_data/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

clusterer_diagnostics_data/2

Extracts the diagnostics metadata from a learned spectral clusterer.

Compilation flags:
`static`

Template:
`clusterer_diagnostics_data(Clusterer,Diagnostics)`
Mode and number of proofs:
`clusterer_diagnostics_data(+compound,-list(compound))` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

clusterer_protocol, *clustering_dataset_protocol*, *kmeans_clusterer*, *linear_algebra*

1.306 sqids

object

1.306.1 `sqids`

Encoding and decoding of short, obfuscated, URL-safe identifiers from lists of non-negative integers, per the Sqids specification, using an atom representation.

Availability:

```
logtalk_load(sqids(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-25

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public sqids(atom)
```

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *decode/2* *decode/3* *default_option/1* *default_options/1* *encode/2* *encode/3* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sqids(Representation), *ids*, *cuid2*, *ksuid*, *nanoid*, *snowflakeid*, *typeid*, *ulid*, *uuid*

object

1.306.2 `sqids(Representation)`

- `Representation` - Text representation for identifiers. Possible values are `atom`, `chars`, and `codes`.

Encoding and decoding of short, obfuscated, URL-safe ids from lists of non-negative integers, per the Sqids specification.

Availability:
`logtalk_load(sqids(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *options*
Uses:
list
type

Remarks:

- Specification: https://github.com/sqids/sqids-spec
- Homepage: https://sqids.org
- Default blocklist: https://github.com/sqids/sqids-blocklist (not bundled; see blocklist/1 option).

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *encode/2*
  - *encode/3*
  - *decode/2*
  - *decode/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

encode/2

Encodes a list of non-negative integers into an identifier using the default options.

Compilation flags:

`static`

Template:

`encode(Numbers,Id)`

Mode and number of proofs:

`encode(+list(integer),-text)` - `one`

Exceptions:

`Numbers` is a variable or a partial list:

`instantiation_error`

`Numbers` is neither a variable nor a list:

`type_error(list(non_negative_integer),Numbers)`

An element `Number` of the list `Numbers` is not an integer:

`type_error(integer,Number)`

An element `Number` of the list `Numbers` of Numbers is a negative integer:

`domain_error(non_negative_integer,Number)`

Too many blocklist collisions occurred while generating the id:

`resource_error(blocklist_retries)`

---

encode/3

Encodes a list of non-negative integers into an id using the given options. Recognized options are alphabet(Atom) (an atom with at least three unique characters; default is the standard 62-character alphabet), min_length(Integer) (an integer between 0 and 255; default is 0), and blocklist(list(atom)) (words that must not occur in the generated id; default is []). If Numbers is empty, Id is unified with the empty atom regardless of the other options.

Compilation flags:

`static`

Template:

```
encode(Numbers,Id,Options)
```

Mode and number of proofs:

```
encode(+list(integer),-text,+list(compound)) - one_or_error
```

Exceptions:

`Numbers` is a variable or a partial list:

```
instantiation_error
```

`Numbers` is neither a variable nor a list:

```
type_error(list(non_negative_integer),Numbers)
```

An element `Number` of the list `Numbers` is not an integer:

```
type_error(integer,Number)
```

An element `Number` of the list `Numbers` of Numbers is a negative integer:

```
domain_error(non_negative_integer,Number)
```

`Options` is a variable or a partial list:

```
instantiation_error
```

`Options` is neither a variable nor a list:

```
type_error(list,Options)
```

An element `Option` of the list `Options` is neither a variable nor a compound term:

```
type_error(compound,Option)
```

An element `Option` of the list `Options` is a compound term but not a valid option:

```
domain_error(option,Option)
```

Too many blocklist collisions occurred while generating the id:

```
resource_error(blocklist_retries)
```

---

decode/2

Decodes an identifier back into a list of non-negative integers using the default options. Equivalent to decode/3 with an empty options list. Numbers is [] if Id is the empty atom or contains a character that is not in the alphabet. Decoding does not check that Id is a canonical (minimal) encoding: ids rejected by encode/2-3 due to the blocklist still decode successfully, as do ids with extraneous minimum-length padding.

Compilation flags:

```
static
```

Template:

```
decode(Id,Numbers)
```

Mode and number of proofs:

```
decode(+text,-list(integer)) - one_or_error
```

Exceptions:

`Id` is a variable:

```
instantiation_error
```

`Id` is neither a variable nor an atom:

```
type_error(atom,Id)
```

---

decode/3

Decodes an identifier back into a list of non-negative integers using the given options. The only recognized option is alphabet(Atom); it must be the same alphabet that was used to encode Id. Numbers is [] if Id is the empty atom or contains a character that is not in the alphabet.

Compilation flags:

`static`

Template:

`decode(Id,Numbers,Options)`

Mode and number of proofs:

`decode(+text,-list(integer),+list(compound))` - `one_or_error`

Exceptions:

`Id` is a variable:

`instantiation_error`

`Id` is neither a variable nor an atom:

`type_error(atom,Id)`

`Options` is a variable or a partial list:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sqids, *ids(Representation,Bytes)*, *cuid2(Representation,Size,Alphabet)*, *ksuid(Representation,Alphabet)*, *nanoid(Representation,Size,Alphabet)*, *snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceB*
typeid(Representation), *ulid(Representation)*, *uuid(Representation)*

1.307 statistics

object

1.307.1 population

Statistical population represented as a list of numbers.

Availability:
`logtalk_load(statistics(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2026-02-20

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *statistics*

Remarks:
(none)

Inherited public predicates:
arithmetic_mean/2 *average_deviation/3* *central_moment/3* *coefficient_of_variation/2* *correlation/3* *covariance/3* *fractile/3* *frequency_distribution/2* *geometric_mean/2* *harmonic_mean/2* *interquartile_range/2* *kurtosis/2* *max/2* *mean_deviation/2* *mean_squared_error/3* *median/2* *median_deviation/2* *min/2* *min_max/3* *min_max_normalization/2* *modes/2* *percentile/3* *product/2* *quartiles/4* *range/2* *rank_correlation/3* *relative_standard_deviation/2* *root_mean_squared_error/3* *skewness/2* *standard_deviation/2* *standard_error/2* *sum/2* *sum_of_squares/2* *trimmed_mean/3* *valid/1* *variance/2* *weighted_mean/3* *z_normalization/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

sample

object

1.307.2 `sample`

Statistical sample represented as a list of numbers.

Availability:
`logtalk_load(statistics(loader))`

Author: Paulo Moura
Version: 1:5:0
Date: 2026-02-20

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *statistics*

Remarks:
(none)

Inherited public predicates:
arithmetic_mean/2 average_deviation/3 central_moment/3 coefficient_of_variation/2 correlation/3 covariance/3 fractile/3 frequency_distribution/2 geometric_mean/2 harmonic_mean/2 interquartile_range/2 kurtosis/2 max/2 mean_deviation/2 mean_squared_error/3 median/2 median_deviation/2 min/2 min_max/3 min_max_normalization/2 modes/2 percentile/3 product/2 quartiles/4 range/2 rank_correlation/3 relative_standard_deviation/2 root_mean_squared_error/3 skewness/2 standard_deviation/2 standard_error/2 sum/2 sum_of_squares/2 trimmed_mean/3 valid/1 variance/2 weighted_mean/3 z_normalization/2

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

population

category

1.307.3 `statistics`

Statistical calculations over a list of numbers.

Availability:
`logtalk_load(statistics(loader))`

Author: Paulo Moura
Version: 1:8:0
Date: 2026-02-20

Compilation flags:
`static`

Implements:
`public` *statisticsp*
Uses:
list
numberlist

Remarks:
(none)

Inherited public predicates:
arithmetic_mean/2 *average_deviation/3* *central_moment/3* *coefficient_of_variation/2* *correlation/3* *covariance/3* *fractile/3* *frequency_distribution/2* *geometric_mean/2* *harmonic_mean/2* *interquartile_range/2* *kurtosis/2* *max/2* *mean_deviation/2* *mean_squared_error/3* *median/2* *median_deviation/2* *min/2* *min_max/3* *min_max_normalization/2* *modes/2* *percentile/3* *product/2* *quartiles/4* *range/2* *rank_correlation/3* *relative_standard_deviation/2* *root_mean_squared_error/3* *skewness/2* *standard_deviation/2* *standard_error/2* *sum/2* *sum_of_squares/2* *trimmed_mean/3* *valid/1* *variance/2* *weighted_mean/3* *z_normalization/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *arithmetic_mean/5*
  - *squares_and_cubes/6*
  - *squares_and_hypers/6*
  - *variance/6*
  - *cross_deviation_sum/6*
  - *sorted_median/3*
  - *sum_of_squares/4*
  - *central_moment_sum/7*
  - *min_max_normalize/4*
  - *count_frequencies/4*
  - *squared_error_sum/6*
  - *compute_ranks/3*
  - *count_less_equal/6*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

arithmetic_mean/5

Auxiliary predicate for computing the arithmetic mean.

Compilation flags:

```
static
```

Template:

```
arithmetic_mean(List,Length0,Length,Sum,Mean)
```

Mode and number of proofs:

```
arithmetic_mean(+list(number),+integer,-integer,+number,-float) - one
```

squares_and_cubes/6

Auxiliary predicate for computing the skewness.

Compilation flags:
`static`

Template:
`squares_and_cubes(List,Mean,Squares0,Squares,Cubes0,Cubes)`
Mode and number of proofs:
`squares_and_cubes(+list(number),+float,+float,-float,+float,-float)` - `one`

---

squares_and_hypers/6

Auxiliary predicate for computing the kurtosis.

Compilation flags:
`static`

Template:
`squares_and_hypers(List,Mean,Squares0,Squares,Hypers0,Hypers)`
Mode and number of proofs:
`squares_and_hypers(+list(number),+float,+float,-float,+float,-float)` - `one`

---

variance/6

Auxiliary predicate for computing the variance.

Compilation flags:
`static`

Template:
`variance(List,Length0,Length,Mean,M20,M2)`
Mode and number of proofs:
`variance(+list(number),+integer,-integer,+float,+float,-float)` - `one`

---

cross_deviation_sum/6

Auxiliary predicate for computing the cross-deviation sum for covariance.

Compilation flags:
```
static
```

Template:
```
cross_deviation_sum(List1,List2,Mean1,Mean2,Sum0,Sum)
```
Mode and number of proofs:
```
cross_deviation_sum(+list(number),+list(number),+float,+float,+float,-float) - one
```

---

sorted_median/3

Auxiliary predicate for computing the median of an already sorted list with known length.

Compilation flags:
```
static
```

Template:
```
sorted_median(Sorted,Length,Median)
```
Mode and number of proofs:
```
sorted_median(+list(number),+integer,-number) - one
```

---

sum_of_squares/4

Auxiliary predicate for computing the sum of squared deviations.

Compilation flags:
```
static
```

Template:
```
sum_of_squares(List,Mean,Sum0,Sum)
```
Mode and number of proofs:
```
sum_of_squares(+list(number),+float,+float,-float) - one
```

---

central_moment_sum/7

Auxiliary predicate for computing central moments.

Compilation flags:

```
static
```

Template:

```
central_moment_sum(List,K,Mean,N0,Sum0,N,Sum)
```

Mode and number of proofs:

```
central_moment_sum(+list(number),+positive_integer,+float,+integer,+float,
-integer,-float) - one
```

---

min_max_normalize/4

Auxiliary predicate for min-max normalization.

Compilation flags:

```
static
```

Template:

```
min_max_normalize(List,Min,Range,NormalizedList)
```

Mode and number of proofs:

```
min_max_normalize(+list(number),+number,+number,-list(float)) - one
```

---

count_frequencies/4

Auxiliary predicate for computing frequency distribution.

Compilation flags:

```
static
```

Template:

```
count_frequencies(List,CurrentValue,CurrentCount,Distribution)
```

Mode and number of proofs:

```
count_frequencies(+list(number),+number,+integer,-list(pair(number,integer))) -
one
```

---

squared_error_sum/6

Auxiliary predicate for computing mean squared error.

Compilation flags:
`static`

Template:
`squared_error_sum(List1,List2,N0,Sum0,N,Sum)`
Mode and number of proofs:
`squared_error_sum(+list(number),+list(number),+integer,+float,-integer,-float)` - `one`

---

compute_ranks/3

Auxiliary predicate for computing ranks of elements in a list.

Compilation flags:
`static`

Template:
`compute_ranks(List,All,Ranks)`
Mode and number of proofs:
`compute_ranks(+list(number),+list(number),-list(float))` - `one`

---

count_less_equal/6

Auxiliary predicate for counting elements less than and equal to a value.

Compilation flags:
`static`

Template:
`count_less_equal(List,Value,Less0,Less,Equal0,Equal)`
Mode and number of proofs:
`count_less_equal(+list(number),+number,+integer,-integer,+integer,-integer)` - `one`

Operators

(none)
protocol

1.307.4 statisticsp

Statistical calculations over a list of numbers protocol.

Availability:
`logtalk_load(statistics(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2026-02-20

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *product/2*
  - *sum/2*
  - *min/2*
  - *max/2*
  - *min_max/3*
  - *range/2*
  - *arithmetic_mean/2*
  - *geometric_mean/2*
  - *harmonic_mean/2*
  - *weighted_mean/3*
  - *median/2*
  - *modes/2*
  - *average_deviation/3*
  - *mean_deviation/2*
  - *median_deviation/2*
  - *standard_deviation/2*
  - *coefficient_of_variation/2*
  - *relative_standard_deviation/2*
  - *skewness/2*

- *kurtosis/2*
- *variance/2*
- *z_normalization/2*
- *fractile/3*
- *percentile/3*
- *quartiles/4*
- *interquartile_range/2*
- *covariance/3*
- *correlation/3*
- *rank_correlation/3*
- *trimmed_mean/3*
- *sum_of_squares/2*
- *central_moment/3*
- *min_max_normalization/2*
- *frequency_distribution/2*
- *standard_error/2*
- *mean_squared_error/3*
- *root_mean_squared_error/3*
- *valid/1*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

product/2

Calculates the product of all list numbers. Fails if the list is empty.

Compilation flags:

```
static
```

Template:

```
product(List,Product)
```

Mode and number of proofs:

```
product(+list(number),-number) - zero_or_one
```

sum/2

Calculates the sum of all list numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`sum(List,Sum)`
Mode and number of proofs:
`sum(+list(number),-number)` - `zero_or_one`

---

min/2

Determines the minimum value in a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`min(List,Minimum)`
Mode and number of proofs:
`min(+list,-number)` - `zero_or_one`

---

max/2

Determines the list maximum value in a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`max(List,Maximum)`
Mode and number of proofs:
`max(+list,-number)` - `zero_or_one`

---

min_max/3

Determines the minimum and maximum values in a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`min_max(List,Minimum,Maximum)`
Mode and number of proofs:
`min_max(+list(number),-number,-number)` - `zero_or_one`

---

range/2

Range is the length of the smallest interval which contains all the numbers in List. Fails if the list is empty.

Compilation flags:
`static`

Template:
`range(List,Range)`
Mode and number of proofs:
`range(+list,-number)` - `zero_or_one`

---

arithmetic_mean/2

Calculates the arithmetic mean of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`arithmetic_mean(List,Mean)`
Mode and number of proofs:
`arithmetic_mean(+list(number),-float)` - `zero_or_one`

---

geometric_mean/2

Calculates the geometric mean of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`geometric_mean(List,Mean)`
Mode and number of proofs:
`geometric_mean(+list(number),-float)` - `zero_or_one`

---

harmonic_mean/2

Calculates the harmonic mean of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`harmonic_mean(List,Mean)`
Mode and number of proofs:
`harmonic_mean(+list(number),-float)` - `zero_or_one`

---

weighted_mean/3

Calculates the weighted mean of a list of numbers. Fails if the list is empty or if the two lists have different lengths. Weights are assumed to be non-negative.

Compilation flags:
`static`

Template:
`weighted_mean(Weights,List,Mean)`
Mode and number of proofs:
`weighted_mean(+list(number),+list(number),-float)` - `zero_or_one`

---

median/2

Calculates the median of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`median(List,Median)`
Mode and number of proofs:
`median(+list(number),-float)` - `zero_or_one`

---

modes/2

Returns the list of modes of a list of numbers in ascending order. Fails if the list is empty.

Compilation flags:
`static`

Template:
`modes(List,Modes)`
Mode and number of proofs:
`modes(+list(number),-list(number))` - `zero_or_one`

---

average_deviation/3

Calculates the average absolute deviation of a list of numbers given a central tendency (e.g., mean, median, or mode). Fails if the list is empty.

Compilation flags:
`static`

Template:
`average_deviation(List,CentralTendency,Deviation)`
Mode and number of proofs:
`average_deviation(+list(number),+float,-float)` - `zero_or_one`

---

mean_deviation/2

Calculates the mean absolute deviation of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`mean_deviation(List,Deviation)`
Mode and number of proofs:
`mean_deviation(+list(number),-float)` - `zero_or_one`

---

median_deviation/2

Calculates the median absolute deviation of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`median_deviation(List,Deviation)`
Mode and number of proofs:
`median_deviation(+list(number),-float)` - `zero_or_one`

---

standard_deviation/2

Calculates the standard deviation of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`standard_deviation(List,Deviation)`
Mode and number of proofs:
`standard_deviation(+list(number),-float)` - `zero_or_one`

---

coefficient_of_variation/2

Calculates the coefficient of variation of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`coefficient_of_variation(List,Coefficient)`
Mode and number of proofs:
`coefficient_of_variation(+list(number),-float)` - `zero_or_one`

---

relative_standard_deviation/2

Calculates the relative standard deviation of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`relative_standard_deviation(List,Percentage)`
Mode and number of proofs:
`relative_standard_deviation(+list(number),-float)` - `zero_or_one`

---

skewness/2

Calculates the (moment) skewness of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`skewness(List,Skewness)`
Mode and number of proofs:
`skewness(+list(number),-float)` - `zero_or_one`

---

kurtosis/2

Calculates the (excess) kurtosis of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`kurtosis(List,Kurtosis)`
Mode and number of proofs:
`kurtosis(+list(number),-float)` - `zero_or_one`

---

variance/2

Calculates the unbiased variance of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`variance(List,Variance)`
Mode and number of proofs:
`variance(+list(number),-float)` - `zero_or_one`

---

z_normalization/2

Normalizes a list of number such that for the resulting list the mean of is close to zero and the standard deviation is close to 1. Fails if the list is empty.

Compilation flags:
`static`

Template:
`z_normalization(List,NormalizedList)`
Mode and number of proofs:
`z_normalization(+list(number),-list(float))` - `zero_or_one`

---

fractile/3

Calculates the smallest value in a list of numbers such that the list elements in its fraction `P` are less or equal to that value (with `P` in the open interval `(0.0, 1.0)`). Fails if the list is empty.

Compilation flags:
`static`

Template:
`fractile(P,List,Fractile)`
Mode and number of proofs:
`fractile(+float,+list(integer),-integer)` - `zero_or_one`
`fractile(+float,+list(float),-float)` - `zero_or_one`

---

percentile/3

Calculates the P-th percentile of a list of numbers (with `P` in the open interval `(0, 100)`). Fails if the list is empty.

Compilation flags:
`static`

Template:
`percentile(P,List,Percentile)`
Mode and number of proofs:
`percentile(+number,+list(number),-number)` - `zero_or_one`

---

quartiles/4

Calculates the quartiles (Q1, Q2, Q3) of a list of numbers. Q2 is the median. Q1 and Q3 are the medians of the lower and upper halves, respectively. Fails if the list has fewer than two elements.

Compilation flags:
`static`

Template:
`quartiles(List,Q1,Q2,Q3)`
Mode and number of proofs:
`quartiles(+list(number),-number,-number,-number)` - `zero_or_one`

---

interquartile_range/2

Calculates the interquartile range (Q3 - Q1) of a list of numbers. Fails if the list has fewer than two elements.

Compilation flags:
```
static
```

Template:
```
interquartile_range(List,IQR)
```
Mode and number of proofs:
```
interquartile_range(+list(number),-number) - zero_or_one
```

---

covariance/3

Calculates the covariance of two lists of numbers. Fails if the lists are empty or have different lengths.

Compilation flags:
```
static
```

Template:
```
covariance(List1,List2,Covariance)
```
Mode and number of proofs:
```
covariance(+list(number),+list(number),-float) - zero_or_one
```

---

correlation/3

Calculates the Pearson correlation coefficient of two lists of numbers. Fails if the lists are empty or have different lengths.

Compilation flags:
```
static
```

Template:
```
correlation(List1,List2,Correlation)
```
Mode and number of proofs:
```
correlation(+list(number),+list(number),-float) - zero_or_one
```

---

rank_correlation/3

Calculates the Spearman rank correlation coefficient of two lists of numbers. Handles ties using average ranks. Fails if the lists are empty or have different lengths.

Compilation flags:
```
static
```

Template:
```
rank_correlation(List1,List2,Correlation)
```
Mode and number of proofs:
```
rank_correlation(+list(number),+list(number),-float) - zero_or_one
```

---

trimmed_mean/3

Calculates the trimmed mean of a list of numbers by removing a fraction of extreme values from both ends (with the fraction in the half-open interval `[0.0, 0.5)`). Fails if the list is empty or if too many elements would be trimmed.

Compilation flags:
```
static
```

Template:
```
trimmed_mean(Fraction,List,Mean)
```
Mode and number of proofs:
```
trimmed_mean(+float,+list(number),-float) - zero_or_one
```

---

sum_of_squares/2

Calculates the sum of squared deviations from the arithmetic mean of a list of numbers. Fails if the list is empty.

Compilation flags:
```
static
```

Template:
```
sum_of_squares(List,Sum)
```
Mode and number of proofs:
```
sum_of_squares(+list(number),-float) - zero_or_one
```

---

central_moment/3

Calculates the K-th central moment of a list of numbers. The K-th central moment is the mean of the deviations from the mean raised to the power K (with `K > 0`). Fails if the list is empty.

Compilation flags:

```
static
```

Template:

```
central_moment(K,List,Moment)
```

Mode and number of proofs:

```
central_moment(+positive_integer,+list(number),-float) - zero_or_one
```

---

min_max_normalization/2

Normalizes a list of numbers to the interval `[0.0, 1.0]` using min-max normalization. Fails if the list is empty or if all values are equal.

Compilation flags:

```
static
```

Template:

```
min_max_normalization(List,NormalizedList)
```

Mode and number of proofs:

```
min_max_normalization(+list(number),-list(float)) - zero_or_one
```

---

frequency_distribution/2

Computes the frequency distribution of a list of numbers, returning a list of `Value-Count` pairs in ascending order of value. Fails if the list is empty.

Compilation flags:

```
static
```

Template:

```
frequency_distribution(List,Distribution)
```

Mode and number of proofs:

```
frequency_distribution(+list(number),-list(pair(number,integer))) - zero_or_one
```

---

standard_error/2

Calculates the standard error of the mean of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`standard_error(List,Error)`
Mode and number of proofs:
`standard_error(+list(number),-float)` - `zero_or_one`

---

mean_squared_error/3

Calculates the mean squared error between two lists of numbers. Fails if the lists are empty or have different lengths.

Compilation flags:
`static`

Template:
`mean_squared_error(List1,List2,Error)`
Mode and number of proofs:
`mean_squared_error(+list(number),+list(number),-float)` - `zero_or_one`

---

root_mean_squared_error/3

Calculates the root mean squared error between two lists of numbers. Fails if the lists are empty or have different lengths.

Compilation flags:
`static`

Template:
`root_mean_squared_error(List1,List2,Error)`
Mode and number of proofs:
`root_mean_squared_error(+list(number),+list(number),-float)` - `zero_or_one`

`valid/1`

Term is a closed list of numbers.

Compilation flags:
`static`

Template:
`valid(Term)`
Mode and number of proofs:
`valid(@nonvar)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

statistics, *sample*, *population*

1.308 stemming

object

1.308.1 `lovins_stemmer(Representation)`

- `Representation` - Word representation. Valid values are `atom`, `codes`, and `chars`.

Lovins stemmer algorithm implementation for English words.

Availability:
`logtalk_load(stemming(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-20

Compilation flags:
`static, context_switching_calls`

Implements:

`public` *stemmer_protocol*

Uses:

list

Remarks:

- Algorithm: The Lovins stemming algorithm (Lovins, 1968) removes the longest suffix from a word using a list of 294 endings, each associated with a condition for removal. It then applies transformation rules to fix spelling.
- Reference: Lovins, J.B. (1968). Development of a stemming algorithm. Mechanical Translation and Computational Linguistics, 11(1-2), 22-31.

Inherited public predicates:

stem/2 *stems/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stemmer_protocol, *porter_stemmer(Representation)*

object

1.308.2 `porter_stemmer(Representation)`

- `Representation` - Word representation. Valid values are `atom`, `codes`, and `chars`.

Porter stemmer algorithm implementation for English words.

Availability:

`logtalk_load(stemming(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-02-05

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stemmer_protocol*

Uses:
integer
list

Remarks:

- Algorithm: The Porter stemming algorithm (Porter, 1980) is a widely used algorithm for reducing English words to their root form by applying a series of rules that remove common suffixes.
- Reference: Porter, M.F. (1980). An algorithm for suffix stripping. Program, 14(3), 130-137.

Inherited public predicates:
stem/2 *stems/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stemmer_protocol, *lovins_stemmer(Representation)*

protocol

1.308.3 stemmer_protocol

Stemmer protocol for reducing words to their stems.

Availability:
`logtalk_load(stemming(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *stem/2*
  - *stems/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

stem/2

Stems a single word, returning its root form.

Compilation flags:
`static`

Template:
`stem(Word,Stem)`
Mode and number of proofs:
`stem(+text,-text)` - `one`

stems/2

Stems a list of words, returning a list of their root forms.

Compilation flags:
```
static
```

Template:
```
stems(Words,Stems)
```
Mode and number of proofs:
```
stems(+list(text),-list(text)) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

porter_stemmer(Representation), *lovins_stemmer(Representation)*

1.309 stomp

object

1.309.1 stomp

Portable STOMP 1.2 (Simple Text Orientated Messaging Protocol) client. Uses the sockets library for TCP communication.

Availability:
```
logtalk_load(stomp(loader))
```

Author: Paulo Moura
Version: 1:2:0
Date: 2026-11-06

Compilation flags:
```
static, context_switching_calls
```

Imports:
```
public options
```

Uses:

list
socket
term_io
user
uuid(Representation)

Remarks:

- Supported backends: ECLiPSe, GNU Prolog, SICStus Prolog, Shumway Prolog, and SWI-Prolog (same as the sockets library).
- Protocol version: Implements STOMP 1.2 specification.
- Heartbeat: Supports heartbeat negotiation. Automatic heartbeat sending is not implemented; use send_heartbeat/1 manually if needed.
- Subscriptions: Supports multiple concurrent subscriptions with unique IDs.
- Transactions: Supports STOMP transactions with BEGIN, COMMIT, and ABORT.
- Frame encoding: Properly encodes/decodes header values according to STOMP 1.2 escaping rules.

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *connect/4*
  - *disconnect/2*
  - *connection_alive/1*
  - *send/4*
  - *subscribe/4*
  - *unsubscribe/3*
  - *receive/3*
  - *ack/3*
  - *nack/3*
  - *begin_transaction/3*
  - *commit_transaction/3*
  - *abort_transaction/3*
  - *send_heartbeat/1*
  - *frame_command/2*
  - *frame_header/3*
  - *frame_headers/2*
  - *frame_body/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

connect/4

Connects to a STOMP server and performs the STOMP handshake. Returns a connection handle for subsequent operations.

Compilation flags:

```
static
```

Template:

```
connect(Host,Port,Connection,Options)
```

Mode and number of proofs:

```
connect(+atom,+integer,--compound,+list) - one_or_error
```

Exceptions:

Connection refused or network error:

```
stomp_error(connection_failed)
```

Server rejected connection:

```
stomp_error(protocol_error(Message))
```

Remarks:

- Option login(Login): Username for authentication.
- Option passcode(Passcode): Password for authentication.
- Option host(VirtualHost): Virtual host name. Defaults to the Host parameter.
- Option heartbeat(ClientMs, ServerMs): Heartbeat timing in milliseconds. Default is 0,0 (no heartbeat).

---

disconnect/2

Gracefully disconnects from the STOMP server. Sends DISCONNECT frame and waits for RECEIPT if requested.

Compilation flags:

```
static
```

Template:

```
disconnect(Connection,Options)
```

Mode and number of proofs:

```
disconnect(+compound,+list) - one_or_error
```

Exceptions:

`Connection` is not a valid STOMP connection handle:

```
error
```

Remarks:

- Option receipt(ReceiptId): Request receipt confirmation. Automatically generated if not specified.

---

connection_alive/1

Checks if the connection is still open and valid.

Compilation flags:
`static`

Template:
`connection_alive(Connection)`
Mode and number of proofs:
`connection_alive(+compound)` - `zero_or_one`

---

send/4

Sends a message to the specified destination.

Compilation flags:
`static`

Template:
`send(Connection,Destination,Body,Options)`
Mode and number of proofs:
`send(+compound,+atom,+term,+list)` - `one_or_error`

Exceptions:
`Connection` is invalid, `Body` cannot be encoded, or the SEND frame cannot be transmitted:
`stomp_error(error)`

Remarks:

- Option content_type(MimeType): MIME type of the body.
- Option content_length(Length): Body length in bytes. Auto-calculated if omitted for atom/string bodies.
- Option transaction(TransactionId): Include message in the named transaction.
- Option receipt(ReceiptId): Request receipt confirmation.
- Option header(Name, Value): Add custom header (can be repeated).

---

subscribe/4

Subscribes to a destination to receive messages.

Compilation flags:
`static`

Template:
`subscribe(Connection,Destination,SubscriptionId,Options)`
Mode and number of proofs:

`subscribe(+compound,+atom,+atom,+list)` - `one_or_error`

Exceptions:

`Connection` is invalid, the acknowledgment mode is unsupported, or the SUBSCRIBE frame cannot be transmitted:

`stomp_error(error)`

Remarks:

- Option ack(Mode): Acknowledgment mode: auto (default), client, or client_individual.

---

unsubscribe/3

Unsubscribes from a destination.

Compilation flags:

`static`

Template:

`unsubscribe(Connection,SubscriptionId,Options)`

Mode and number of proofs:

`unsubscribe(+compound,+atom,+list)` - `one_or_error`

Exceptions:

`Connection` is invalid or the UNSUBSCRIBE frame cannot be transmitted:

`stomp_error(error)`

---

receive/3

Receives a frame from the server. Returns MESSAGE, RECEIPT, or ERROR frames.

Compilation flags:

`static`

Template:

`receive(Connection,Frame,Options)`

Mode and number of proofs:

`receive(+compound,-compound,+list)` - `zero_or_one_or_error`

Exceptions:

`Connection` is invalid or the input stream does not contain a valid STOMP frame within the requested timeout:

`error`

Remarks:

- Option timeout(Milliseconds): Timeout in milliseconds. 0 for non-blocking, -1 for infinite wait. Default is -1.

---

ack/3

Acknowledges receipt of a message.

Compilation flags:
`static`

Template:
`ack(Connection,AckId,Options)`
Mode and number of proofs:
`ack(+compound,+atom,+list)` - `one_or_error`

Exceptions:
`Connection` is invalid or the ACK frame cannot be transmitted:
`stomp_error(error)`

Remarks:

- Option transaction(TransactionId): Include acknowledgment in the named transaction.

---

nack/3

Negatively acknowledges a message (tells server the message was not consumed).

Compilation flags:
`static`

Template:
`nack(Connection,AckId,Options)`
Mode and number of proofs:
`nack(+compound,+atom,+list)` - `one_or_error`

Exceptions:
`Connection` is invalid or the NACK frame cannot be transmitted:
`stomp_error(error)`

Remarks:

- Option transaction(TransactionId): Include negative acknowledgment in the named transaction.

---

begin_transaction/3

Begins a new transaction.

Compilation flags:
`static`

Template:
`begin_transaction(Connection,TransactionId,Options)`
Mode and number of proofs:
`begin_transaction(+compound,+atom,+list)` - `one_or_error`

Exceptions:
`Connection` is invalid or the BEGIN frame cannot be transmitted:
`stomp_error(error)`

---

commit_transaction/3

Commits a transaction, making all its operations permanent.

Compilation flags:
`static`

Template:
`commit_transaction(Connection,TransactionId,Options)`
Mode and number of proofs:
`commit_transaction(+compound,+atom,+list)` - `one_or_error`

Exceptions:
`Connection` is invalid or the COMMIT frame cannot be transmitted:
`stomp_error(error)`

---

abort_transaction/3

Aborts a transaction, rolling back all its operations.

Compilation flags:
`static`

Template:
`abort_transaction(Connection,TransactionId,Options)`
Mode and number of proofs:
`abort_transaction(+compound,+atom,+list)` - `one_or_error`

Exceptions:

`Connection` is invalid or the ABORT frame cannot be transmitted:
`stomp_error(error)`

---

send_heartbeat/1

Sends a heartbeat (EOL) to the server to keep the connection alive.

Compilation flags:
`static`

Template:
`send_heartbeat(Connection)`
Mode and number of proofs:
`send_heartbeat(+compound)` - `one_or_error`

Exceptions:
`Connection` is invalid or the heartbeat cannot be written to the output stream:
`stomp_error(error)`

---

frame_command/2

Extracts the command from a frame.

Compilation flags:
`static`

Template:
`frame_command(Frame,Command)`
Mode and number of proofs:
`frame_command(+compound,-atom)` - `one`

---

frame_header/3

Extracts a header value from a frame. Fails if header is not present.

Compilation flags:
`static`

Template:
`frame_header(Frame,HeaderName,Value)`
Mode and number of proofs:
`frame_header(+compound,+atom,-atom)` - `zero_or_one`

---

frame_headers/2

Extracts all headers from a frame as a list of Name-Value pairs.

Compilation flags:
```
static
```

Template:
```
frame_headers(Frame,Headers)
```
Mode and number of proofs:
```
frame_headers(+compound,-list) - one
```

---

frame_body/2

Extracts the body from a frame. Returns empty atom if no body.

Compilation flags:
```
static
```

Template:
```
frame_body(Frame,Body)
```
Mode and number of proofs:
```
frame_body(+compound,-term) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.310 stop_words

object

1.310.1 stop_words(Representation,Language)

- `Representation` - Word representation. Valid values are `atom`, `codes`, and `chars`.
- `Language` - Object implementing the `stop_words_language_protocol` protocol.

Stop-word enumeration and filtering for a text representation and language provider.

Availability:
`logtalk_load(stop_words(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-05

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_protocol*
Uses:
string(Representation)

Remarks:
(none)

Inherited public predicates:
exclude/2 *is_stop_word/1* *stop_word/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words_protocol, *stop_words_language_protocol*

protocol

1.310.2 stop_words_language_protocol

Stop-word language provider protocol.

Availability:
`logtalk_load(stop_words(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *stop_word/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`stop_word/1`

Enumerates canonical lowercase stop-word atoms.

Compilation flags:
`static`

Template:
`stop_word(Word)`
Mode and number of proofs:
`stop_word(-atom)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

stop_words_protocol, *stop_words(Representation,Language)*

protocol

1.310.3 `stop_words_protocol`

Stop-word enumeration and filtering protocol.

Availability:
`logtalk_load(stop_words(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-05

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *stop_word/1*
  - *is_stop_word/1*
  - *exclude/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

stop_word/1

Enumerates canonical stop words using the configured text representation.

Compilation flags:
`static`

Template:
`stop_word(Word)`
Mode and number of proofs:
`stop_word(?text)` - `zero_or_more`

---

is_stop_word/1

True if the word, after ASCII lowercasing, is a stop word.

Compilation flags:
`static`

Template:
`is_stop_word(Word)`
Mode and number of proofs:
`is_stop_word(+text)` - `zero_or_one`

`exclude/2`

Removes stop words, preserving the representation, value, and order of the remaining words.

Compilation flags:
`static`

Template:
`exclude(Words,Filtered)`
Mode and number of proofs:
`exclude(+list(text),-list(text))` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

stop_words_language_protocol, *stop_words(Representation,Language)*

object

1.310.4 `stopwords_af`

Stop-word facts for the af language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-af
- Source commit: 7bc0179a79af05dd1e57108168cd5e50c5f994c8
- Generated entries: 51.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.5 `stopwords_ar`

Stop-word facts for the ar language code.

Availability:

```
logtalk_load(stop_words(loader))
```

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ar
- Source commit: 4a421ac9875b325909df7e4b7392c31eae047795
- Generated entries: 480.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.6 stopwords_bg

Stop-word facts for the bg language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-bg
- Source commit: e9a4b791692ef3f6b89b3544ea9398a544793aee
- Generated entries: 259.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.7 `stopwords_bn`

Stop-word facts for the bn language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-bn
- Source commit: 943a761b656e33f2e1475f34d997e545161536d8
- Generated entries: 398.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.8 `stopwords_br`

Stop-word facts for the br language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-br
- Source commit: 4b6a22c44e469d57873a943e003bc6cf4b76bd9c
- Generated entries: 1203.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.9 `stopwords_ca`

Stop-word facts for the ca language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ca

- Source commit: 222bb5691f90586016c1a3a8342e199ce7b3e399
- Generated entries: 278.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.10 `stopwords_cs`

Stop-word facts for the cs language code.

Availability:

```
logtalk_load(stop_words(loader))
```

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:

```
static, context_switching_calls
```

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-cs
- Source commit: e93e103f19297e60f69535e41dd216352c363698
- Generated entries: 423.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.11 `stopwords_da`

Stop-word facts for the da language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-da
- Source commit: 591e16c01f6679b73eb7f0e625bc2a9879d54a86
- Generated entries: 170.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.12 `stopwords_de`

Stop-word facts for the de language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-de
- Source commit: d3826f11cd729d31b491aca27d9fdd0a36984885
- Generated entries: 620.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.13 `stopwords_el`

Stop-word facts for the el language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-el
- Source commit: d9e49e4c835570d4425db832832b7e834013ecf5
- Generated entries: 847.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.14 `stopwords_en`

Stop-word facts for the en language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-en
- Source commit: ccc8898188850d8fb019d5f69c14a6635c3bd115
- Source corrections: Removed four malformed non-ASCII reflexive-pronoun entries.
- Generated entries: 1294.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.15 `stopwords_eo`

Stop-word facts for the eo language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-eo

- Source commit: 7cf2e8022c15818771d92c281e7c4756655fdc3b
- Generated entries: 173.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.16 `stopwords_es`

Stop-word facts for the es language code.

Availability:

```
logtalk_load(stop_words(loader))
```

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:

```
static, context_switching_calls
```

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-es
- Source commit: 622c17231fa0e164026de886f2bb8b2e5f0ac825
- Generated entries: 732.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.17 `stopwords_et`

Stop-word facts for the et language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-et
- Source commit: 1ba777413664f4639d0c24511b90763cc88481a2
- Generated entries: 35.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.18 stopwords_eu

Stop-word facts for the eu language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-eu
- Source commit: c9606313d4de34c7cacd25bd08fb3c800a50e2e3
- Generated entries: 98.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.19 `stopwords_fa`

Stop-word facts for the fa language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-fa
- Source commit: ca084a92b2bed8ecfcfbc668368ce90d4dc48422
- Generated entries: 799.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.20 `stopwords_fi`

Stop-word facts for the fi language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-fi
- Source commit: 81f4bbbe99870b0d07bd7deedb87fc66a595dc2a
- Generated entries: 847.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.21 `stopwords_fr`

Stop-word facts for the fr language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-fr

- Source commit: 069f556be89f2f08de2f1b8313084bb4610f30b4
- Generated entries: 691.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.22 `stopwords_ga`

Stop-word facts for the ga language code.

Availability:

```
logtalk_load(stop_words(loader))
```

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:

```
static, context_switching_calls
```

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ga
- Source commit: 699a27ec708829ffe1d1c3b31ab5a700e83ddd09
- Generated entries: 109.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.23 `stopwords_gl`

Stop-word facts for the gl language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-gl
- Source commit: 232aef6362e93cf6b3d5f741fff856ae0e56f25d
- Generated entries: 160.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.24 `stopwords_ha`

Stop-word facts for the ha language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ha
- Source commit: 8fbf04b5967ad5f88322da42826c7673efda2e2b
- Generated entries: 39.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.25 `stopwords_he`

Stop-word facts for the he language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-he
- Source commit: 79a1e86b6c3800c09210e939ef640b5e4af86d83
- Generated entries: 194.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.26 `stopwords_hi`

Stop-word facts for the hi language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-hi
- Source commit: fae0a092c0fb8058f9c2470005467ee697b4763b
- Generated entries: 225.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.27 `stopwords_hr`

Stop-word facts for the hr language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-hr

- Source commit: 56d5bfcc645f25aa822b86ac4b9d63c55c14f4a4
- Generated entries: 179.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.28 `stopwords_hu`

Stop-word facts for the hu language code.

Availability:

```
logtalk_load(stop_words(loader))
```

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-hu
- Source commit: 0b0faacb8cbde083488966c63b5b0e389563ec14
- Generated entries: 789.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.29 `stopwords_hy`

Stop-word facts for the hy language code.

Availability:

`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-hy
- Source commit: a381d40b0cfc970535caa1bcb0495430b19fd255
- Generated entries: 45.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.30 `stopwords_id`

Stop-word facts for the id language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-id
- Source commit: 6109494eb4cfebd75c59bd832848374d02f4e907
- Generated entries: 758.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.31 `stopwords_it`

Stop-word facts for the it language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-it
- Source commit: 89fa17a2fd94debd7f52526cedce2c3d519c8378
- Generated entries: 632.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.32 `stopwords_ja`

Stop-word facts for the ja language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ja
- Source commit: 5a000f6a62f9e3a12f436f36d168e2fcd2fb1878
- Generated entries: 134.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.33 `stopwords_ko`

Stop-word facts for the ko language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ko

- Source commit: bb9e03d893c62d13bce6344523d3d9e76aa5f289
- Generated entries: 679.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.34 `stopwords_ku`

Stop-word facts for the ku language code.

Availability:

```
logtalk_load(stop_words(loader))
```

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:

```
static, context_switching_calls
```

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ku
- Source commit: 2a9d6483eef65231b3e63ef3f237c85d81f52ea0
- Generated entries: 62.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.35 `stopwords_la`

Stop-word facts for the la language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-la
- Source commit: 8bd6439e794e88744a2972ba73f094a19b073121
- Generated entries: 49.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.36 `stopwords_lt`

Stop-word facts for the lt language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-lt
- Source commit: 569e38174b182186283860ffe23966c344763b46
- Generated entries: 474.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.37 `stopwords_lv`

Stop-word facts for the lv language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-lv
- Source commit: db396c48049e75172da56d1b99b90d6e7370d56f
- Generated entries: 161.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.38 `stopwords_mr`

Stop-word facts for the mr language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-mr
- Source commit: bd01f486167e32ea9f25d5425f13e30d07c87a7d
- Generated entries: 99.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.39 `stopwords_ms`

Stop-word facts for the ms language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ms

- Source commit: afc1c238e98d6e671caeea45955c9b292dc9aec6
- Generated entries: 475.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.40 `stopwords_nl`

Stop-word facts for the nl language code.

Availability:

```
logtalk_load(stop_words(loader))
```

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-nl
- Source commit: 72a51c02c275c0dbc877e783aaaaa5440d199542
- Generated entries: 413.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.41 `stopwords_no`

Stop-word facts for the no language code.

Availability:

`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-no
- Source commit: bdc5473085ba0d75a3efb7e6c152faa8f795d3c3
- Generated entries: 221.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.42 `stopwords_pl`

Stop-word facts for the pl language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-pl
- Source commit: 723fe9b896c5acf35d1d5cc725f012ce2309291f
- Generated entries: 329.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.43 `stopwords_pt`

Stop-word facts for the pt language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-pt
- Source commit: 84249b7ca37005f8cd19df70d86f44dc1f1936a5
- Generated entries: 560.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.44 `stopwords_ro`

Stop-word facts for the ro language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ro
- Source commit: e05061f5296b57bebb7b06a3894d92e9e91e2157
- Generated entries: 434.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.45 `stopwords_ru`

Stop-word facts for the ru language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ru

- Source commit: e1779aae6fb351669a978d41e51939a57e7ef2a6
- Generated entries: 559.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.46 `stopwords_sk`

Stop-word facts for the sk language code.

Availability:

```
logtalk_load(stop_words(loader))
```

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:

```
static, context_switching_calls
```

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-sk
- Source commit: bd3c61983b761f7aed676de3c857313982de123e
- Generated entries: 418.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.47 `stopwords_sl`

Stop-word facts for the sl language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-sl
- Source commit: 5474e42f16556ef97f1dbe45fdeab46234b0ea94
- Generated entries: 446.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.48 `stopwords_so`

Stop-word facts for the so language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-so
- Source commit: f67bd3bd239e7156c6c6aecb478a1c935d632a98
- Generated entries: 30.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.49 `stopwords_st`

Stop-word facts for the st language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-st
- Source commit: 3934239b2a6ec28c9713373363e8d652a090456d
- Generated entries: 31.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.50 `stopwords_sv`

Stop-word facts for the sv language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-sv
- Source commit: 2c46aa5773816a85ed69118acbf05649f9c47390
- Generated entries: 418.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.51 `stopwords_sw`

Stop-word facts for the sw language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-sw

- Source commit: 624fd53c78bd0adee14d798014590f24b1ec8c54
- Generated entries: 74.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.52 `stopwords_th`

Stop-word facts for the th language code.

Availability:

```
logtalk_load(stop_words(loader))
```

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-th
- Source commit: 7fae6c6722c68dd27cbd9205fe67b1d5cdc2a8c3
- Generated entries: 115.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.53 `stopwords_tl`

Stop-word facts for the tl language code.

Availability:

`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-tl
- Source commit: 39fe5dc7e1ac148aa2e57b041f98e17d3741d640
- Generated entries: 147.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.54 `stopwords_tr`

Stop-word facts for the tr language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-tr
- Source commit: 06ece8deadbc0876a3d0188cfcbbddba89df475b
- Generated entries: 504.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.55 `stopwords_uk`

Stop-word facts for the uk language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-uk
- Source commit: 83b24b92ff9f8af0bbad349522c783f9d5a28b88
- Generated entries: 73.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.56 `stopwords_ur`

Stop-word facts for the ur language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-ur
- Source commit: eff1b4c5e3fc56ecc4ae197fc8f2bac043a87ea3
- Generated entries: 517.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.57 `stopwords_vi`

Stop-word facts for the vi language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-vi

- Source commit: 0bf911ed74c5eb15bb6b1ca3b20f0ee390e193e1
- Generated entries: 645.

Inherited public predicates:

stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.58 `stopwords_yo`

Stop-word facts for the yo language code.

Availability:

```
logtalk_load(stop_words(loader))
```

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:

```
static, context_switching_calls
```

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-yo
- Source commit: ab33b417b40b2677c035a790da8d3e87e9174166
- Generated entries: 60.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.59 `stopwords_zh`

Stop-word facts for the zh language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-zh
- Source commit: 0f62ec9b9061b9142ed87fa6b39d386945c7cf47
- Generated entries: 794.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

object

1.310.60 `stopwords_zu`

Stop-word facts for the zu language code.

Availability:
`logtalk_load(stop_words(loader))`

Author: Gene Diaz
Version: 1:0:0
Date: 2026-09-05

Copyright: Copyright (c) 2016 Gene Diaz
License: MIT

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *stop_words_language_protocol*

Remarks:

- Source: https://github.com/stopwords-iso/stopwords-zu
- Source commit: 5f1e34361fe0d5fba6eb8659150e880b205bcfb0
- Generated entries: 29.

Inherited public predicates:
stop_word/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

stop_words(Representation,Language)

1.311 string_distance

object

1.311.1 `string_distance(Representation)`

- `Representation` - String representation. Valid values are `atom`, `codes`, and `chars`.

String distance predicates.

Availability:
`logtalk_load(string_distance(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Uses:
integer
list
set

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *levenshtein/3*
  - *damerau_levenshtein/3*
  - *hamming/3*
  - *jaro/3*
  - *jaro_winkler/3*

- edit_similarity/3
- edit_similarity/4
- longest_common_subsequence_length/3
- longest_common_subsequence/3
- longest_common_substring/3
- cosine_similarity/3
- jaccard_index/3
- soundex/2
- soundex_match/2
- nysiis/2
- nysiis_match/2
- metaphone/2
- metaphone_match/2
- double_metaphone/3
- double_metaphone_match/2
- caverphone/2
- caverphone_match/2

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

levenshtein/3

Computes the Levenshtein distance between two strings.

Compilation flags:

`static`

Template:

`levenshtein(String1,String2,Distance)`

`String1` - First input string.

`String2` - Second input string.

`Distance` - Minimum number of single-character edits (insertions, deletions, substitutions) to transform `String1` into `String2`.

Mode and number of proofs:

`levenshtein(+text,+text,-integer)` - `one`

damerau_levenshtein/3

Computes the Damerau-Levenshtein distance between two strings.

Compilation flags:

`static`

Template:

`damerau_levenshtein(String1,String2,Distance)`

`String1` - First input string.

`String2` - Second input string.

`Distance` - Minimum number of edits (insertions, deletions, substitutions, and adjacent transpositions) to transform `String1` into `String2`.

Mode and number of proofs:

`damerau_levenshtein(+text,+text,-integer)` - `one`

---

hamming/3

Computes the Hamming distance between two strings of equal length. Fails if the strings differ in length.

Compilation flags:

`static`

Template:

`hamming(String1,String2,Distance)`

`String1` - First input string.

`String2` - Second input string (must have the same length as `String1`).

`Distance` - Number of positions at which the corresponding characters differ.

Mode and number of proofs:

`hamming(+text,+text,-integer)` - `zero_or_one`

---

jaro/3

Computes the Jaro similarity score between two strings.

Compilation flags:

`static`

Template:

`jaro(String1,String2,Similarity)`

`String1` - First input string.

`String2` - Second input string.

`Similarity` - A value between 0.0 (completely different) and 1.0 (identical), based on matching characters and transpositions.

Mode and number of proofs:

`jaro(+text,+text,-float)` - `one`

---

jaro_winkler/3

Computes the Jaro-Winkler similarity score between two strings.

Compilation flags:
`static`

Template:
`jaro_winkler(String1,String2,Similarity)`
`String1` - First input string.
`String2` - Second input string.
`Similarity` - A value between 0.0 and 1.0. Extends Jaro similarity with a prefix bonus: strings sharing a common prefix are scored higher.

Mode and number of proofs:
`jaro_winkler(+text,+text,-float)` - `one`

---

edit_similarity/3

Computes the edit similarity score between two strings using Levenshtein distance.

Compilation flags:
`static`

Template:
`edit_similarity(String1,String2,Similarity)`
`String1` - First input string.
`String2` - Second input string.
`Similarity` - A value between 0.0 and 1.0 computed as 1 - (edit distance / max length of the two strings).

Mode and number of proofs:
`edit_similarity(+text,+text,-float)` - `one`

---

edit_similarity/4

Computes the edit similarity score between two strings using the given algorithm.

Compilation flags:
`static`

Template:
`edit_similarity(Algorithm,String1,String2,Similarity)`

`Algorithm` - Edit distance algorithm. Valid values are `levenshtein`, `damerau_levenshtein`, `hamming`, and `longest_common_subsequence`.
`String1` - First input string.
`String2` - Second input string.
`Similarity` - A value between 0.0 and 1.0 computed as 1 - (edit distance / max length of the two strings).

Mode and number of proofs:

`edit_similarity(+atom,+text,+text,-float)` - `one`

`longest_common_subsequence_length/3`

Computes the length of the Longest Common Subsequence between two strings.

Compilation flags:

`static`

Template:

`longest_common_subsequence_length(String1,String2,Length)`
`String1` - First input string.
`String2` - Second input string.
`Length` - Length of the longest subsequence common to both strings (characters need not be contiguous).

Mode and number of proofs:

`longest_common_subsequence_length(+text,+text,-integer)` - `one`

`longest_common_subsequence/3`

Computes the Longest Common Subsequence itself between two strings.

Compilation flags:

`static`

Template:

`longest_common_subsequence(String1,String2,Subsequence)`
`String1` - First input string.
`String2` - Second input string.
`Subsequence` - The longest subsequence common to both strings (characters need not be contiguous). If multiple exist, one is returned nondeterministically.

Mode and number of proofs:

`longest_common_subsequence(+text,+text,-atom)` - `one`

---

longest_common_substring/3

Computes the longest contiguous common substring between two strings.

Compilation flags:
`static`

Template:
`longest_common_substring(String1,String2,Substring)`
`String1` - First input string.
`String2` - Second input string.
`Substring` - The longest contiguous substring shared by both strings. If multiple exist, one is returned nondeterministically.

Mode and number of proofs:
`longest_common_substring(+text,+text,-atom)` - `one`

---

cosine_similarity/3

Computes the cosine similarity between two token lists.

Compilation flags:
`static`

Template:
`cosine_similarity(Tokens1,Tokens2,Similarity)`
`Tokens1` - First token list (e.g., list of words or character n-grams).
`Tokens2` - Second token list.
`Similarity` - A value between 0.0 and 1.0 representing the cosine of the angle between the two token vectors.

Mode and number of proofs:
`cosine_similarity(+list(text),+list(text),-float)` - `one`

---

jaccard_index/3

Computes the Jaccard index (similarity) between two token lists.

Compilation flags:
`static`

Template:
`jaccard_index(Tokens1,Tokens2,Index)`
`Tokens1` - First token list (e.g., list of words or character n-grams).
`Tokens2` - Second token list.
`Index` - A value between 0.0 (no overlap) and 1.0 (identical sets), computed as `|intersection| / |union|`.

Mode and number of proofs:
`jaccard_index(+list(text),+list(text),-float)` - `one`

---

soundex/2

Computes the Soundex phonetic encoding for a string.

Compilation flags:
`static`

Template:
`soundex(String,Encoding)`
`String` - Input string (typically a name).
`Encoding` - A four-character Soundex code representing the phonetic encoding.

Mode and number of proofs:
`soundex(+text,-atom)` - `one`

---

soundex_match/2

Succeeds if two strings share the same Soundex code.

Compilation flags:
`static`

Template:
`soundex_match(String1,String2)`
`String1` - First input string.
`String2` - Second input string.

Mode and number of proofs:
`soundex_match(+text,+text)` - `one`

---

nysiis/2

Computes the NYSIIS phonetic key for a string.

Compilation flags:
`static`

Template:
`nysiis(String,Encoding)`
`String` - Input string (typically a name).
`Encoding` - The NYSIIS phonetic encoding, limited to six characters.

Mode and number of proofs:
`nysiis(+text,-atom)` - `one`

---

nysiis_match/2

Succeeds if two strings share the same NYSIIS key.

Compilation flags:
`static`

Template:
`nysiis_match(String1,String2)`
`String1` - First input string.
`String2` - Second input string.

Mode and number of proofs:
`nysiis_match(+text,+text)` - `zero_or_one`

---

metaphone/2

Computes the Metaphone phonetic key for a string.

Compilation flags:
`static`

Template:
`metaphone(String,Encoding)`
`String` - Input string (typically a name).
`Encoding` - The Metaphone phonetic encoding, a more accurate phonetic encoding than Soundex.

Mode and number of proofs:
`metaphone(+text,-atom)` - `one`

---

metaphone_match/2

Succeeds if two strings share the same Metaphone key.

Compilation flags:
`static`

Template:
`metaphone_match(String1,String2)`
`String1` - First input string.
`String2` - Second input string.

Mode and number of proofs:
`metaphone_match(+text,+text)` - `one`

---

double_metaphone/3

Computes the Double Metaphone encoding of a text, returning both primary and alternative encodings.

Compilation flags:

`static`

Template:

`double_metaphone(Text,Primary,Alternative)`

`Text` - Input string (typically a name).

`Primary` - Primary Double Metaphone encoding.

`Alternative` - Alternative Double Metaphone encoding.

Mode and number of proofs:

`double_metaphone(+text,-atom,-atom)` - `one`

---

double_metaphone_match/2

Succeeds if the Double Metaphone encodings of two texts match (either primary or alternative encodings).

Compilation flags:

`static`

Template:

`double_metaphone_match(String1,String2)`

`String1` - First input string.

`String2` - Second input string.

Mode and number of proofs:

`double_metaphone_match(+text,+text)` - `zero_or_one`

---

caverphone/2

Computes the Caverphone 2.0 phonetic encoding of a text.

Compilation flags:

`static`

Template:

`caverphone(Text,Encoding)`

`Text` - Input string (typically a name).

`Encoding` - The Caverphone 2.0 phonetic encoding, limited to ten characters.

Mode and number of proofs:

`caverphone(+text,-atom)` - `one`

---

caverphone_match/2

Succeeds if two strings share the same Caverphone 2.0 key.

Compilation flags:
`static`

Template:
`caverphone_match(String1,String2)`
`String1` - First input string.
`String2` - Second input string.

Mode and number of proofs:
`caverphone_match(+text,+text)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.312 strings

object

1.312.1 string(Representation)

- `Representation` - String representation. Valid values are `atom`, `codes`, and `chars`.

String manipulation predicates supporting different string representations.

Availability:
`logtalk_load(strings(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-04-17

Compilation flags:
`static, context_switching_calls`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *atom_string/2*
  - *number_string/2*
  - *string_chars/2*
  - *string_codes/2*
  - *string_concat/3*
  - *string_length/2*
  - *sub_string/5*
  - *string_upper/2*
  - *string_lower/2*
  - *split_string/4*
  - *atomics_to_string/2*
  - *atomics_to_string/3*
  - *trim/2*
  - *trim/3*
  - *trim_left/2*
  - *trim_left/3*
  - *trim_right/2*
  - *trim_right/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

atom_string/2

Converts between an atom and a string.

Compilation flags:
`static`

Template:
`atom_string(Atom,String)`

Mode and number of proofs:
`atom_string(+atom,?text)` - `zero_or_one`
`atom_string(-atom,+text)` - `zero_or_one`

number_string/2

Converts between a number and a string. Fails if the string does not represent a valid number.

Compilation flags:
```
static
```

Template:
```
number_string(Number,String)
```
Mode and number of proofs:
```
number_string(+number,?text) - zero_or_one
number_string(-number,+text) - zero_or_one
```

---

string_chars/2

Converts between a string and a list of characters.

Compilation flags:
```
static
```

Template:
```
string_chars(String,Chars)
```
Mode and number of proofs:
```
string_chars(+text,?list(character)) - zero_or_one
string_chars(-text,+list(character)) - zero_or_one
```

---

string_codes/2

Converts between a string and a list of character codes.

Compilation flags:
```
static
```

Template:
```
string_codes(String,Codes)
```
Mode and number of proofs:
```
string_codes(+text,?list(character_code)) - zero_or_one
string_codes(-text,+list(character_code)) - zero_or_one
```

---

string_concat/3

Concatenates two strings.

Compilation flags:
`static`

Template:
`string_concat(String1,String2,String3)`
Mode and number of proofs:
`string_concat(+text,+text,?text)` - `zero_or_one`
`string_concat(?text,?text,+text)` - `zero_or_more`

---

string_length/2

Returns the length of a string.

Compilation flags:
`static`

Template:
`string_length(String,Length)`
Mode and number of proofs:
`string_length(+text,?integer)` - `zero_or_one`

---

sub_string/5

Extracts a substring from a string.

Compilation flags:
`static`

Template:
`sub_string(String,Before,Length,After,SubString)`
Mode and number of proofs:
`sub_string(+text,?integer,?integer,?integer,?text)` - `zero_or_more`

---

string_upper/2

Converts a string to uppercase (ASCII only).

Compilation flags:
`static`

Template:
`string_upper(String,UpperString)`
Mode and number of proofs:
`string_upper(+text,?text)` - `zero_or_one`

---

string_lower/2

Converts a string to lowercase (ASCII only).

Compilation flags:
`static`

Template:
`string_lower(String,LowerString)`
Mode and number of proofs:
`string_lower(+text,?text)` - `zero_or_one`

---

split_string/4

Decomposes `String` into `SubStrings` according to separators `SepChars` and padding characters `PadChars`. The string is split at the separators, and any padding characters around the resulting sub-strings are removed. Characters in both `SepChars` and `PadChars` are treated as separators where sequences count as one separator, and are ignored at string boundaries.

Compilation flags:
`static`

Template:
`split_string(String,SepChars,PadChars,SubStrings)`
Mode and number of proofs:
`split_string(+text,+text,+text,-list(text))` - `one`

---

atomics_to_string/2

Concatenates the atomic terms in `List` into `String`. The list may contain numbers, atoms, and strings (in the current representation).

Compilation flags:
`static`

Template:
`atomics_to_string(List,String)`
Mode and number of proofs:
`atomics_to_string(++list(atomic),-text)` - `one`

---

atomics_to_string/3

Concatenates the atomic terms in `List` into `String`, with `Separator` inserted between each element. The list may contain numbers, atoms, and strings (in the current representation).

Compilation flags:
`static`

Template:
`atomics_to_string(List,Separator,String)`
Mode and number of proofs:
`atomics_to_string(++list(atomic),+text,-text)` - `one`

---

trim/2

Trims string by deleting all leading and trailing whitespace.

Compilation flags:
`static`

Template:
`trim(String,Trimmed)`
Mode and number of proofs:
`trim(+text,-text)` - `one`

---

trim/3

Trims string by deleting all occurrences of the characters in `Elements` from the beginning and end of the string.

Compilation flags:
`static`

Template:
`trim(String,Elements,Trimmed)`
Mode and number of proofs:
`trim(+text,+text,-text)` - `one`

---

trim_left/2

Trims string by deleting all leading whitespace.

Compilation flags:
`static`

Template:
`trim_left(String,Trimmed)`
Mode and number of proofs:
`trim_left(+text,-text)` - `one`

---

trim_left/3

Trims string by deleting all occurrences of the characters in `Elements` from the beginning of the string.

Compilation flags:
`static`

Template:
`trim_left(String,Elements,Trimmed)`
Mode and number of proofs:
`trim_left(+text,+text,-text)` - `one`

---

trim_right/2

Trims string by deleting all trailing whitespace.

Compilation flags:
`static`

Template:
`trim_right(String,Trimmed)`
Mode and number of proofs:
`trim_right(+text,-text)` - `one`

---

trim_right/3

Trims string by deleting all occurrences of the characters in `Elements` from the end of the string.

Compilation flags:
`static`

Template:
`trim_right(String,Elements,Trimmed)`
Mode and number of proofs:
`trim_right(+text,+text,-text)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.313 subsequences

object

1.313.1 subsequences

Implementation of subsequence operations over lists.

Availability:
`logtalk_load(subsequences(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-28

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *subsequences_protocol*
Uses:
fast_random(Algorithm)
list

Remarks:
(none)

Inherited public predicates:
alternating_subsequence/2 *alternating_subsequences/2* *common_subsequence/3* *common_subsequences/3* *count_distinct_subsequences/3* *count_subsequences/2* *init/2* *init1/2* *init_tail/2* *init_tails/2* *inits/2* *inits1/2* *is_prefix_of/2* *is_subsequence_of/2* *is_suffix_of/2* *k_distinct_subsequence/3* *k_distinct_subsequences/3* *longest_common_increasing_subsequence/3* *longest_common_subsequence/3* *longest_decreasing_subsequence/2* *longest_increasing_subsequence/2* *longest_repeating_subsequence/2* *nonempty_subsequence/2* *nonempty_subsequences/2* *power_set/2* *proper_subsequence/2* *random_subsequence/2* *sliding_window/3* *subsequence/2* *subsequence/3* *subsequence_at_indices/3* *subsequence_length/2* *subsequences/2* *subsequences/3* *subsequences_with_min_span/3* *subslices/2* *tail/2* *tail1/2* *tails/2* *tails1/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.313.2 `subsequences_protocol`

Protocol for subsequence operations over lists.

Availability:

```
logtalk_load(subsequences(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-26

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

- Generation operations: Predicates for generating all subsequences or variants thereof.
- Ordering variants: Predicates that support an additional Order argument (default, lexicographic, or shortlex) for controlling output order.
- Searching and matching: Predicates for finding specific subsequences with desired properties.
- Prefix and suffix operations: Predicates for checking and finding prefixes and suffixes.
- Contiguous subsequences: Predicates for working with contiguous subsequences (subslices, sliding windows).
- Random selection: Predicates for randomly selecting subsequences.
- Constrained operations: Predicates for generating subsequences with specific constraints.
- Utility predicates: Helper predicates for subsequence operations.

Inherited public predicates:

(none)

- *Public predicates*

- *subsequences/2*
- *subsequence/2*
- *subsequences/3*
- *subsequence/3*
- *nonempty_subsequences/2*
- *nonempty_subsequence/2*
- *power_set/2*
- *inits/2*
- *init/2*
- *tails/2*
- *tail/2*
- *inits1/2*
- *init1/2*
- *tails1/2*
- *tail1/2*
- *init_tails/2*
- *init_tail/2*
- *longest_common_subsequence/3*
- *longest_increasing_subsequence/2*
- *longest_decreasing_subsequence/2*
- *longest_common_increasing_subsequence/3*
- *longest_repeating_subsequence/2*
- *is_subsequence_of/2*
- *proper_subsequence/2*
- *subsequence_at_indices/3*
- *common_subsequences/3*
- *common_subsequence/3*
- *count_distinct_subsequences/3*
- *is_prefix_of/2*
- *is_suffix_of/2*
- *subslices/2*
- *sliding_window/3*
- *random_subsequence/2*
- *subsequences_with_min_span/3*
- *alternating_subsequences/2*
- *alternating_subsequence/2*
- *k_distinct_subsequences/3*
- *k_distinct_subsequence/3*
- *count_subsequences/2*
- *subsequence_length/2*

- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

subsequences/2

Generates all subsequences of a list using default order. A subsequence maintains the relative order of elements but need not be contiguous. The empty list is included.

Compilation flags:

```
static
```

Template:

```
subsequences(List,Subsequences)
```

Mode and number of proofs:

```
subsequences(+list,-list) - one
```

Examples:

All subsequences

```
subsequences([a,b,c],Subsequences)
Subsequences=[[],[a],[b],[a,b],[c],[a,c],[b,c],[a,b,c]]
```

---

subsequence/2

True iff the second argument is a subsequence of the first argument. Subsequences of a list using default order. A subsequence maintains the relative order of elements but need not be contiguous. The empty list is included.

Compilation flags:

```
static
```

Template:

```
subsequence(List,Subsequence)
```

Mode and number of proofs:

```
subsequence(+list,-list) - one_or_more
```

Examples:

A subsequence

```
subsequence([1,2],Subsequence)
Subsequence=[]
```

---

subsequences/3

Generates all subsequences of a list with specified ordering: `default` (as naturally produced), `lexicographic`, or `shortlex` (by length first, then lexicographically).

Compilation flags:
`static`

Template:
`subsequences(List,Order,Subsequences)`
Mode and number of proofs:
`subsequences(+list,+atom,-list)` - `one`

Examples:
Shortlex order
```
subsequences([a,b],shortlex,Subsequences)
Subsequences=[[],[a],[b],[a,b]]
```

---

subsequence/3

True iff the third argument is a subsequence of the first argument with specified ordering: `default` (as naturally produced), `lexicographic`, or `shortlex` (by length first, then lexicographically).

Compilation flags:
`static`

Template:
`subsequence(List,Order,Subsequence)`
Mode and number of proofs:
`subsequence(+list,+atom,-list)` - `one_or_more`

Examples:
Shortlex order
```
subsequence([a,b],shortlex,Subsequence)
Subsequence=[]
```

---

nonempty_subsequences/2

Generates all non-empty subsequences of a list.

Compilation flags:
`static`

Template:
`nonempty_subsequences(List,Subsequences)`

Mode and number of proofs:

```
nonempty_subsequences(+list,-list) - one
```

Examples:

Non-empty subsequences

```
nonempty_subsequences([a,b],Subsequences)
Subsequences=[[a],[b],[a,b]]
```

---

nonempty_subsequence/2

True iff the second argument is a non-empty subsequence of the first argument.

Compilation flags:

```
static
```

Template:

```
nonempty_subsequence(List,Subsequence)
```

Mode and number of proofs:

```
nonempty_subsequence(+list,-list) - one_or_more
```

Examples:

Non-empty subsequence

```
nonempty_subsequence([a,b],Subsequence)
Subsequence=[a]
```

---

power_set/2

Generates the power set of a list (all possible subsequences). Alias for subsequences/2 when first argument is ground.

Compilation flags:

```
static
```

Template:

```
power_set(List,PowerSet)
```

Mode and number of proofs:

```
power_set(+list,-list) - one
```

Examples:

Power set

```
power_set([a,b],PowerSet)
PowerSet=[[],[a],[b],[a,b]]
```

---

inits/2

Generates all initial segments (prefixes) of a list, shortest first. Includes the empty list.

Compilation flags:
```
static
```

Template:
```
inits(List,Inits)
```
Mode and number of proofs:
```
inits(+list,-list) - one
```

Examples:
All prefixes
```
inits([a,b,c],Inits)
Inits=[[],[a],[a,b],[a,b,c]]
```

---

init/2

True iff the second argument is one of the initial segments (prefixes) of a list.

Compilation flags:
```
static
```

Template:
```
init(List,Inits)
```
Mode and number of proofs:
```
init(+list,-term) - zero_or_more
init(+list,+term) - zero_or_one
```

Examples:
Check prefix
```
init([a,b,c],[a,b])
true
```

---

tails/2

Generates all final segments (suffixes) of a list, longest first. Includes the empty list.

Compilation flags:
```
static
```

Template:
```
tails(List,Tails)
```
Mode and number of proofs:

```
tails(+list,-list) - one
```

Examples:

All suffixes

```
tails([a,b,c],Tails)
Tails=[[a,b,c],[b,c],[c],[]]
```

---

tail/2

True iff the second argument is one of the final segments (suffixes) of a list.

Compilation flags:

```
static
```

Template:

```
tail(List,Tails)
```

Mode and number of proofs:

```
tail(+list,-term) - zero_or_more
tail(+list,+term) - zero_or_one
```

Examples:

Check suffix

```
tail([a,b,c],[b,c])
true
```

---

inits1/2

Generates all non-empty initial segments (prefixes) of a list, shortest first.

Compilation flags:

```
static
```

Template:

```
inits1(List,Inits)
```

Mode and number of proofs:

```
inits1(+list,-list) - one
```

Examples:

Non-empty prefixes

```
inits1([a,b,c],Inits)
Inits=[[a],[a,b],[a,b,c]]
```

---

init1/2

True iff the second argument is a non-empty initial segment (prefix) of a list, shortest first.

Compilation flags:
`static`

Template:
`init1(List,Init)`
Mode and number of proofs:
`init1(+list,-term)` - `one_or_more`

Examples:
Non-empty prefix
```
init1([a,b,c],Init)
Init=[a]
```

---

tails1/2

Generates all non-empty final segments (suffixes) of a list, longest first.

Compilation flags:
`static`

Template:
`tails1(List,Tails)`
Mode and number of proofs:
`tails1(+list,-list)` - `one`

Examples:
Non-empty suffix
```
tails1([a,b,c],Tails)
Tails=[[a,b,c],[b,c],[c]]
```

---

tail1/2

True iff the second argument is a non-empty final segment (suffix) of a list, longest first.

Compilation flags:
`static`

Template:
`tail1(List,Tail)`
Mode and number of proofs:
`tail1(+list,-list)` - `one`

Examples:

Non-empty suffix

```
tail1([a,b,c],Tail)
Tail=[a,b,c]
```

---

init_tails/2

Generates all pairs of initial and final segments. Each pair Init-Tail represents a split of the list where Init+Tail equals the original list. When the second argument is bound, checks if it is a valid split.

Compilation flags:

```
static
```

Template:

```
init_tails(List,InitTailPairs)
```

Mode and number of proofs:

```
init_tails(+list,-list) - one
```

Examples:

All splits

```
init_tails([a,b],InitTailPairs)
InitTailPairs=['-'([],[a,b]),'-'([a],[b]),'-'([a,b],[])]
```

---

init_tail/2

True iff (Init,Tail) represents a split of the list where Init+Tail equals the original list.

Compilation flags:

```
static
```

Template:

```
init_tail(List,InitTailPairs)
```

Mode and number of proofs:

```
init_tail(+list,-term) - one_or_more
```

Examples:

Check split

```
init_tail([a,b,c],'-'([a],[b,c]))
true
```

---

longest_common_subsequence/3

Finds the longest common subsequence (LCS) between two lists. Uses dynamic programming.

Compilation flags:
```
static
```

Template:
```
longest_common_subsequence(List1,List2,LCS)
```
Mode and number of proofs:
```
longest_common_subsequence(+list,+list,-list) - one
```

Examples:
LCS example
```
longest_common_subsequence([a,b,c,d,e],[a,c,e,f],LCS)
LCS=[a,c,e]
```

---

longest_increasing_subsequence/2

Finds the longest strictly increasing subsequence in a list. Elements must be comparable.

Compilation flags:
```
static
```

Template:
```
longest_increasing_subsequence(List,LIS)
```
Mode and number of proofs:
```
longest_increasing_subsequence(+list,-list) - one
```

Examples:
LIS example
```
longest_increasing_subsequence([3,1,4,1,5,9,2,6],LIS)
LIS=[1,4,5,9]
```

---

longest_decreasing_subsequence/2

Finds the longest strictly decreasing subsequence in a list. Elements must be comparable.

Compilation flags:
```
static
```

Template:
```
longest_decreasing_subsequence(List,LDS)
```
Mode and number of proofs:
```
longest_decreasing_subsequence(+list,-list) - one
```

Examples:

LDS example

```
longest_decreasing_subsequence([9,5,2,8,3,1],LDS)
LDS=[9,5,2,1]
```

---

longest_common_increasing_subsequence/3

Finds the longest subsequence that is both common to two lists and strictly increasing.

Compilation flags:

```
static
```

Template:

```
longest_common_increasing_subsequence(List1,List2,LCIS)
```

Mode and number of proofs:

```
longest_common_increasing_subsequence(+list,+list,-list) - one
```

Examples:

LCIS example

```
longest_common_increasing_subsequence([1,4,2,5],[4,1,3,5],LCIS)
LCIS=[1,5]
```

---

longest_repeating_subsequence/2

Finds the longest subsequence that appears at least twice in the list (at different positions).

Compilation flags:

```
static
```

Template:

```
longest_repeating_subsequence(List,LRS)
```

Mode and number of proofs:

```
longest_repeating_subsequence(+list,-list) - one
```

Examples:

LRS example

```
longest_repeating_subsequence([a,a,b,a,b],LRS)
LRS=[a,b]
```

---

is_subsequence_of/2

Checks if the first list is a subsequence of the second list. All elements must occur in order.

Compilation flags:

```
static
```

Template:

```
is_subsequence_of(Subsequence,List)
```

Mode and number of proofs:

```
is_subsequence_of(+list,+list) - zero_or_one
```

Examples:

Valid subsequence

```
is_subsequence_of([a,c],[a,b,c])
true
```

Invalid subsequence

```
is_subsequence_of([c,a],[a,b,c])
false
```

---

proper_subsequence/2

Checks if the first list is a proper subsequence of the second list (i.e., subsequence and not equal).

Compilation flags:

```
static
```

Template:

```
proper_subsequence(Subsequence,List)
```

Mode and number of proofs:

```
proper_subsequence(+list,+list) - zero_or_one
```

Examples:

Proper subsequence

```
proper_subsequence([a,c],[a,b,c])
true
```

Not proper (equal lists)

```
proper_subsequence([a,b,c],[a,b,c])
false
```

---

subsequence_at_indices/3

Extracts a subsequence using a strictly increasing list of 1-based indices.

Compilation flags:
`static`

Template:
`subsequence_at_indices(List,Indices,Subsequence)`
Mode and number of proofs:
`subsequence_at_indices(+list,+list,-list)` - `zero_or_one`

Examples:
Indices selection
```
subsequence_at_indices([a,b,c,d],[1,3],Subsequence)
Subsequence=[a,c]
```

---

common_subsequences/3

Generates all subsequences that are common to both lists.

Compilation flags:
`static`

Template:
`common_subsequences(List1,List2,CommonSubsequences)`
Mode and number of proofs:
`common_subsequences(+list,+list,-list)` - `one`

Examples:
All common subsequences
```
common_subsequences([a,b,c],[a,c,d],CommonSubsequences)
CommonSubsequences=[[a,c],[a],[c],[]]
```

---

common_subsequence/3

True iff the third argument is a common subsequence of both lists.

Compilation flags:
`static`

Template:
`common_subsequence(List1,List2,CommonSubsequence)`
Mode and number of proofs:
`common_subsequence(+list,+list,-list)` - `one_or_more`

Examples:

Check common subsequence

```
common_subsequence([a,b,c],[a,c,d],[a,c])
true
```

---

count_distinct_subsequences/3

Counts the number of distinct occurrences of a pattern as a subsequence in a list.

Compilation flags:

```
static
```

Template:

```
count_distinct_subsequences(Pattern,List,Count)
```

Mode and number of proofs:

```
count_distinct_subsequences(+list,+list,-integer) - one
```

Examples:

Count occurrences

```
count_distinct_subsequences([a,b],[a,a,b,b],Count)
Count=4
```

---

is_prefix_of/2

Checks if the first list is a prefix of the second list.

Compilation flags:

```
static
```

Template:

```
is_prefix_of(Prefix,List)
```

Mode and number of proofs:

```
is_prefix_of(+list,+list) - zero_or_one
```

Examples:

Valid prefix

```
is_prefix_of([a,b],[a,b,c])
true
```

Invalid prefix

```
is_prefix_of([b,c],[a,b,c])
false
```

---

is_suffix_of/2

Checks if the first list is a suffix of the second list.

Compilation flags:

```
static
```

Template:

```
is_suffix_of(Suffix,List)
```

Mode and number of proofs:

```
is_suffix_of(+list,+list) - zero_or_one
```

Examples:

Valid suffix

```
is_suffix_of([b,c],[a,b,c])
true
```

Invalid suffix

```
is_suffix_of([a,b],[a,b,c])
false
```

---

subslices/2

Generates all contiguous non-empty subslices (sublists) of a list.

Compilation flags:

```
static
```

Template:

```
subslices(List,Subslice)
```

Mode and number of proofs:

```
subslices(+list,-list) - one
subslices(+list,?list) - zero_or_more
```

Examples:

All subslices

```
subslices([a,b,c],Subslice)
Subslice=[[a],[a,b],[a,b,c],[b],[b,c],[c]]
```

---

sliding_window/3

Generates all contiguous windows of size N from a list. Fails if N is larger than the list length.

Compilation flags:
```
static
```

Template:
```
sliding_window(N,List,Window)
```
Mode and number of proofs:
```
sliding_window(+integer,+list,-list) - one
sliding_window(+integer,+list,?list) - zero_or_more
```

Examples:

Windows of size 2
```
sliding_window(2,[a,b,c,d],Window)
Window=[[a,b],[b,c],[c,d]]
```

---

random_subsequence/2

Randomly selects one subsequence uniformly from all 2^N possible subsequences.

Compilation flags:
```
static
```

Template:
```
random_subsequence(List,Subsequence)
```
Mode and number of proofs:
```
random_subsequence(+list,-list) - one
```

Examples:

Random subsequence
```
random_subsequence([a,b,c],Subsequence)
Subsequence=[a,c]
```

---

subsequences_with_min_span/3

Generates subsequences where consecutive elements are at least MinSpan positions apart in the original list.

Compilation flags:
```
static
```

Template:
```
subsequences_with_min_span(MinSpan,List,Subsequence)
```
Mode and number of proofs:

```
subsequences_with_min_span(+integer,+list,-list) - one
subsequences_with_min_span(+integer,+list,?list) - zero_or_more
```

Examples:

Min span of 2

```
subsequences_with_min_span(2,[a,b,c,d],Subsequence)
Subsequence=[a,c]
```

---

alternating_subsequences/2

Generates all subsequences that alternate between increasing and decreasing (or vice versa). Elements must be comparable.

Compilation flags:

```
static
```

Template:

```
alternating_subsequences(List,AlternatingSubsequences)
```

Mode and number of proofs:

```
alternating_subsequences(+list,-list) - one
```

Examples:

All alternating subsequences

```
alternating_subsequences([1,3,2,4],AlternatingSubsequences)
AlternatingSubsequences=[[1,2],[1,3],[1,4],[2,3],[2,4],[3,4]]
```

---

alternating_subsequence/2

True iff the second argument is a subsequence that alternates between increasing and decreasing (or vice versa). Elements must be comparable.

Compilation flags:

```
static
```

Template:

```
alternating_subsequence(List,AlternatingSubsequence)
```

Mode and number of proofs:

```
alternating_subsequence(+list,-list) - one_or_more
```

Examples:

Alternating

```
alternating_subsequence([1,3,2,4],AlternatingSubsequence)
AlternatingSubsequence=[[1,2],[1,3],[1,4],[2,3],[2,4],[3,4]]
```

---

k_distinct_subsequences/3

Generates all K-element subsequences where all elements are distinct (no duplicates in the subsequence itself).

Compilation flags:

```
static
```

Template:

```
k_distinct_subsequences(K,List,DistinctSubsequences)
```

Mode and number of proofs:

```
k_distinct_subsequences(+integer,+list,-list) - one
```

Examples:

All distinct only

```
k_distinct_subsequences(2,[a,a,b],DistinctSubsequences)
DistinctSubsequences=[[a,b],[a,c],[b,c]]
```

---

k_distinct_subsequence/3

True iff the third argument is a subsequence of the first argument that is a K-element subsequence where all elements are distinct (no duplicates in the subsequence itself).

Compilation flags:

```
static
```

Template:

```
k_distinct_subsequence(K,List,DistinctSubsequence)
```

Mode and number of proofs:

```
k_distinct_subsequence(+integer,+list,-list) - one
```

Examples:

A distinct only

```
k_distinct_subsequence(2,[a,a,b],DistinctSubsequence)
DistinctSubsequence=[a,b]
```

---

count_subsequences/2

Counts the total number of subsequences (always 2^N for a list of length N).

Compilation flags:

```
static
```

Template:

```
count_subsequences(List,Count)
```

Mode and number of proofs:

```
count_subsequences(+list,-integer) - one
```

Examples:

Count

```
count_subsequences([a,b,c],Count)
Count=8
```

---

subsequence_length/2

Returns the length of a subsequence (same as list length, provided for consistency).

Compilation flags:

```
static
```

Template:

```
subsequence_length(Subsequence,Length)
```

Mode and number of proofs:

```
subsequence_length(+list,-integer) - one
```

Examples:

Length

```
subsequence_length([a,b,c],Length)
Length=3
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.314 svr_regression

object

1.314.1 svr_regression

Support vector regression (SVR) using an epsilon-insensitive dual margin model with linear, polynomial, and radial basis function kernels, trained with a Pegasos-style dual subgradient descent loop. Learns from a dataset object implementing the `regression_dataset_protocol` protocol and returns a regressor term that can be used for prediction and exported as predicate clauses.

Availability:

```
logtalk_load(svr_regression(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-13

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *regressor_common*

Uses:

format
linear_algebra
list
pairs
type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_regressor/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *print_regressor/1* *regressor_options/2* *valid_option/1* *valid_options/1* *valid_regressor/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

regression_dataset_protocol, *kernel_svm_classifier*, *linear_svm_classifier*, *linear_regression*, *ridge_regression*, *lasso_regression*

1.315 tabu_search

object

1.315.1 `tabu_search(Problem)`

- `Problem` - Problem object implementing `tabu_search_problem_protocol`.

Tabu search optimization algorithm using the Xoshiro128++ random number generator via `fast_random(xoshiro128pp)`.

Availability:
`logtalk_load(tabu_search(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *tabu_search(Problem,fast_random(xoshiro128pp))*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *run/2* *run/3* *run/4* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

tabu_search(Problem,Random), *tabu_search_problem_protocol*

object

1.315.2 `tabu_search(Problem,Random)`

- `Problem` - Problem object implementing `tabu_search_problem_protocol`.
- `Random` - Random number generator object providing `between/3`, `randomize/1`, `permutation/2`, and `sample/4` predicates.

Tabu search optimization algorithm. Parameterized by a problem object implementing the `tabu_search_problem_protocol` protocol and by a random number generator algorithm for the `fast_random` library. Minimizes energy with optional problem-defined tabu keys, tenure, aspiration, restart states, neighbor generation, stopping, and progress reporting.

Availability:

`logtalk_load(tabu_search(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-11-19

Compilation flags:

`static, context_switching_calls`

Imports:

`public` *options*

Uses:

list

type

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *run/2*
  - *run/3*
  - *run/4*
- *Protected predicates*
- *Private predicates*
  - *select_candidate/10*
  - *require_neighborhood/2*
  - *restart_candidate/7*
  - *state_key/3*
  - *resolve_tenure/6*
- *Operators*

Public predicates

run/2

Runs the tabu search algorithm using default options and returns the best state found and its energy.

Compilation flags:
`static`

Template:
`run(BestState,BestEnergy)`
Mode and number of proofs:
`run(-nonvar,-number)` - `one`

Exceptions:
An implemented output hook fails or returns an invalid value:
`domain_error(tabu_hook_result,Result)`

---

run/3

Runs the tabu search algorithm using the given options and returns the best state found and its energy.

Compilation flags:
`static`

Template:
`run(BestState,BestEnergy,Options)`
Mode and number of proofs:
`run(-nonvar,-number,+list(compound))` - `one`

Exceptions:

An option is invalid:

```
domain_error(option,Option)
```

Exhaustive mode requires implemented enumeration:

```
existence_error(procedure,Problem::neighbors/2)
```

An implemented output hook fails or returns an invalid value:

```
domain_error(tabu_hook_result,Result)
```

Remarks:

- `max_steps(N)` option: Maximum number of iterations per cycle (default: `10000`).
- `tabu_tenure(T)` option: Fixed lifetime in selection steps (default: `7`). Ignored when a range or a problem `tabu_tenure/4` implementation is present.
- `tabu_tenure_range(Min, Max)` option: Uniform random tenure in the inclusive integer range. Overrides fixed tenure; ignored when the problem implements `tabu_tenure/4`.
- `candidates(N)` option: Number of candidate neighbors examined per iteration (default: `20`).
- `exhaustive(Boolean)` option: Evaluates the full `neighbors/2` list in source order when true, ignoring the candidate limit (default: `false`). Requires implemented enumeration.
- `updates(N)` option: Number of progress reports during the run. Set to `0` to disable. Progress is reported by calling `progress/5` on the problem object (default: `0`).
- `seed(S)` option: Positive integer seed for the random number generator, enabling reproducible runs (default: none).
- `restarts(N)` option: Number of additional cycles (default: `0`). Clears memory and uses `restart_state/2` when implemented, otherwise the global best state.

`run/4`

Runs the tabu search algorithm using the given options, returns the best state found and its energy, and returns run statistics.

Compilation flags:

```
static
```

Template:

```
run(BestState,BestEnergy,Statistics,Options)
```

Mode and number of proofs:

```
run(-nonvar,-number,-list(compound),+list(compound)) - one
```

Exceptions:

An option is invalid:

```
domain_error(option,Option)
```

Exhaustive mode requires implemented enumeration:

```
existence_error(procedure,Problem::neighbors/2)
```

An implemented output hook fails or returns an invalid value:

```
domain_error(tabu_hook_result,Result)
```

Remarks:

- Statistics list: A list of `Key(Value)` pairs: `steps(N)` is the number of steps executed, `acceptances(A)` is the number of accepted moves, `improvements(I)` is the number of moves that improved the best energy, and `final_tabu_size(S)` is the number of non-expired tabu entries at termination.

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

select_candidate/10

Selects the best admissible candidate using the cached run policy.

Compilation flags:

```
static
```

Template:

```
select_candidate(Policy,State,Energy,BestEnergy,Tabu,Step,Candidates,Neighbor,
NeighborEnergy,Accepted)
```

Mode and number of proofs:

```
select_candidate(+compound,+nonvar,+number,+number,+list,+non_negative_integer,
+positive_integer,-nonvar,-number,-boolean) - one
```

Exceptions:

Enumeration fails in exhaustive mode:

```
domain_error(tabu_hook_result,neighbors/2)
```

A key hook returns an invalid result:

```
domain_error(tabu_hook_result,Result)
```

---

require_neighborhood/2

Checks the implementation required for exhaustive evaluation.

Compilation flags:

```
static
```

Template:

```
require_neighborhood(Exhaustive,Problem)
```

Mode and number of proofs:

```
require_neighborhood(+boolean,+object_identifier) - one
```

Exceptions:

Enumeration is not implemented:

```
existence_error(procedure,Problem::neighbors/2)
```

---

restart_candidate/7

Resolves a restart state and updates the global best independently.

Compilation flags:
`static`

Template:
`restart_candidate(Policy,Best,BestEnergy,State,Energy,NewBest,NewBestEnergy)`
Mode and number of proofs:
`restart_candidate(+compound,+nonvar,+number,-nonvar,-number,-nonvar,-number)` - `one`

Exceptions:
The restart hook fails or returns an invalid state:
`domain_error(tabu_hook_result,Result)`

---

state_key/3

Resolves an optional tabu key without copying the state.

Compilation flags:
`static`

Template:
`state_key(KeyHook,State,Key)`
Mode and number of proofs:
`state_key(+boolean,+nonvar,-nonvar)` - `one`

Exceptions:
The key hook fails or returns an unbound key:
`domain_error(tabu_hook_result,Result)`

---

resolve_tenure/6

Resolves tenure with hook, range, then fixed-option precedence.

Compilation flags:
`static`

Template:
`resolve_tenure(TenureHook,Specification,Step,BestEnergy,CurrentEnergy,Tenure)`
Mode and number of proofs:
`resolve_tenure(+boolean,+compound,+non_negative_integer,+number,+number,`
`-non_negative_integer)` - `one`

Exceptions:
The tenure hook fails or returns an invalid value:
`domain_error(tabu_hook_result,Result)`

Operators

(none)

See also

tabu_search(Problem), *tabu_search_problem_protocol*

protocol

1.315.3 tabu_search_problem_protocol

Protocol for tabu search problems, with optional tabu keys, tenure, aspiration, restart diversification, delta-energy generation, neighborhood enumeration, stopping, and progress reporting.

Availability:
`logtalk_load(tabu_search(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2026-10-10

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *initial_state/1*
  - *neighbor_state/2*
  - *neighbor_state/3*
  - *neighbors/2*
  - *tabu_key/2*
  - *aspiration/3*
  - *tabu_tenure/4*

    - *restart_state/2*
    - *state_energy/2*
    - *stop_condition/3*
    - *progress/5*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

initial_state/1

Returns an initial state for the optimization problem.

Compilation flags:
`static`

Template:
`initial_state(State)`
Mode and number of proofs:
`initial_state(-nonvar)` - `one`

---

neighbor_state/2

Generates a neighboring state from the given state. This is the most problem-specific predicate and its definition determines the quality of the search. Used both for candidate sampling and (when `neighbors/2` is not defined) as the sole neighborhood operator.

Compilation flags:
`static`

Template:
`neighbor_state(State,Neighbor)`
Mode and number of proofs:
`neighbor_state(+nonvar,-nonvar)` - `one`

---

neighbor_state/3

Generates a neighboring state and returns the energy change (delta) directly, avoiding a full energy recomputation. Optional. When not defined by the problem, the algorithm calls `neighbor_state/2` and `state_energy/2` instead.

Compilation flags:

```
static
```

Template:

```
neighbor_state(State,Neighbor,DeltaEnergy)
```

Mode and number of proofs:

```
neighbor_state(+nonvar,-nonvar,-number) - one
```

---

neighbors/2

Optionally returns the complete neighborhood. With `exhaustive(true)`, an implementation must succeed for every visited state, including empty neighborhoods, and all candidates are evaluated in source order. Otherwise the candidate limit controls sampling and failure falls back to neighbor generation.

Compilation flags:

```
static
```

Template:

```
neighbors(State,Neighbors)
```

Mode and number of proofs:

```
neighbors(+nonvar,-list(nonvar)) - zero_or_one
```

Exceptions:

Exhaustive mode has no implemented enumeration predicate:

```
existence_error(procedure,Problem::neighbors/2)
```

Enumeration fails in exhaustive mode:

```
domain_error(tabu_hook_result,neighbors/2)
```

---

tabu_key/2

Optionally returns a stable tabu-equivalence key without instantiating the state. Absent implementations use the state itself. Keys are compared using strict term identity; shared variables retain identity, while fresh variables do not provide stable equivalence.

Compilation flags:

```
static
```

Template:

```
tabu_key(State,Key)
```

Mode and number of proofs:

```
tabu_key(+nonvar,-nonvar) - one
```

Exceptions:

An implemented hook fails:

```
domain_error(tabu_hook_result,tabu_key/2)
```

An implemented hook returns an unbound key:

```
domain_error(tabu_hook_result,tabu_key/2-Key)
```

---

aspiration/3

Optionally permits an otherwise tabu contender. Success permits the move; failure rejects it even when it improves the global best. Absent implementations use strict best-so-far aspiration. Called only for tabu candidates that could win selection; implementations should be pure and must not instantiate the candidate.

Compilation flags:

```
static
```

Template:

```
aspiration(CandidateState,CandidateEnergy,BestEnergy)
```

Mode and number of proofs:

```
aspiration(+nonvar,+number,+number) - zero_or_one
```

---

tabu_tenure/4

Optionally returns tenure once per accepted move, using the zero-based global selection step and pre-move best and current energies. Overrides fixed and ranged options. Zero skips insertion without erasing earlier active entries; existing expiration steps never change.

Compilation flags:

```
static
```

Template:

```
tabu_tenure(Step,BestEnergy,CurrentEnergy,Tenure)
```

Mode and number of proofs:

```
tabu_tenure(+non_negative_integer,+number,+number,-non_negative_integer) - one
```

Exceptions:

An implemented hook fails:

```
domain_error(tabu_hook_result,tabu_tenure/4)
```

An implemented hook returns an invalid tenure:

```
domain_error(tabu_hook_result,tabu_tenure/4-Tenure)
```

---

restart_state/2

Optionally returns a restart state without instantiating the global best state. Called only between cycles. Its energy is recomputed and may update the best, without incrementing move statistics. Absent implementations restart from the best with its cached energy.

Compilation flags:

```
static
```

Template:

```
restart_state(BestState,RestartState)
```

Mode and number of proofs:

```
restart_state(+nonvar,-nonvar) - one
```

Exceptions:

An implemented hook fails:

```
domain_error(tabu_hook_result,restart_state/2)
```

An implemented hook returns an unbound state:

```
domain_error(tabu_hook_result,restart_state/2-RestartState)
```

---

state_energy/2

Computes the energy (cost) of the given state. The algorithm minimizes this value.

Compilation flags:

```
static
```

Template:

```
state_energy(State,Energy)
```

Mode and number of proofs:

```
state_energy(+nonvar,-number) - one
```

---

stop_condition/3

True when the search should stop given the current step, best energy found so far, and current energy. Optional. When not defined by the problem, the search runs until the maximum number of steps is reached.

Compilation flags:

```
static
```

Template:

```
stop_condition(Step,BestEnergy,CurrentEnergy)
```

Mode and number of proofs:

```
stop_condition(+non_negative_integer,+number,+number) - zero_or_one
```

---

progress/5

Called with completed global steps and the actual current energy. Optional. The acceptance and improvement rates are computed since the preceding report, or since the cycle began for its first report. A zero-step interval has zero rates. When reporting is enabled, each cycle has a final report that replaces any coincident periodic report.

Compilation flags:
`static`

Template:
`progress(Step,BestEnergy,CurrentEnergy,AcceptanceRate,ImprovementRate)`
Mode and number of proofs:
`progress(+non_negative_integer,+number,+number,+number,+number)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

tabu_search(Problem)

1.316 term_io

object

1.316.1 term_io

Term input/output from/to atom, chars, and codes.

Availability:
`logtalk_load(term_io(loader))`

Author: Paulo Moura
Version: 1:3:1
Date: 2026-11-04

Compilation flags:
`static, context_switching_calls`

Implements:

`public` *term_io_protocol*

Uses:

os

Remarks:

(none)

Inherited public predicates:

format_to_atom/3 *format_to_chars/3* *format_to_chars/4* *format_to_codes/3* *format_to_codes/4* *read_from_atom/2* *read_from_chars/2* *read_from_codes/2* *read_term_from_atom/3* *read_term_from_chars/3* *read_term_from_chars/4* *read_term_from_codes/3* *read_term_from_codes/4* *with_output_to/2* *write_term_to_atom/3* *write_term_to_chars/3* *write_term_to_chars/4* *write_term_to_codes/3* *write_term_to_codes/4* *write_to_atom/2* *write_to_chars/2* *write_to_codes/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - `temporary_file_/1`
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`temporary_file_/1`

Logtalk session and `term_io` specific temporary file path.

Compilation flags:

`dynamic`

Template:

`temporary_file_(Path)`

Mode and number of proofs:

`temporary_file_(-atom)` - `one`

Operators

(none)

protocol

1.316.2 term_io_protocol

Predicates for term input/output from/to atom, chars, and codes. The predicates are declared as synchronized when the library is compiled using a backend supporting threads.

Availability:

```
logtalk_load(term_io(loader))
```

Author: Paulo Moura
Version: 1:3:1
Date: 2026-06-15

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

- Portability notes: To keep calls to these library predicates portable, use only standard read/write options and specify output formats using atoms.

Inherited public predicates:

(none)

- *Public predicates*
  - *read_term_from_atom/3*
  - *read_from_atom/2*
  - *read_term_from_chars/3*
  - *read_term_from_chars/4*
  - *read_from_chars/2*
  - *read_term_from_codes/3*
  - *read_term_from_codes/4*
  - *read_from_codes/2*
  - *write_term_to_atom/3*
  - *write_to_atom/2*
  - *write_term_to_chars/3*
  - *write_term_to_chars/4*
  - *write_to_chars/2*
  - *write_term_to_codes/3*
  - *write_term_to_codes/4*
  - *write_to_codes/2*

      - *format_to_atom/3*
      - *format_to_chars/3*
      - *format_to_chars/4*
      - *format_to_codes/3*
      - *format_to_codes/4*
      - *with_output_to/2*
    - *Protected predicates*
    - *Private predicates*
    - *Operators*

Public predicates

read_term_from_atom/3

Reads a term from an atom using the given read options. A period at the end of the atom is optional. Valid options are those supported by the standard `read_term/3` predicate.

Compilation flags:
`static, synchronized`

Template:
`read_term_from_atom(Atom,Term,Options)`

Mode and number of proofs:
`read_term_from_atom(+atom,-term,+list(read_option))` - `one_or_error`

Exceptions:
`Atom` is a variable:
`instantiation_error`
`Atom` is neither a variable nor an atom:
`type_error(atom,Atom)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(read_option,Option)`

read_from_atom/2

Reads a term from an atom using default read options. Shorthand for `read_term_from_atom(Atom,Term,[])`. A period at the end of the atom is optional.

Compilation flags:
`static`

Template:
`read_from_atom(Atom,Term)`
Mode and number of proofs:
`read_from_atom(+atom,-term)` - `one_or_error`

Exceptions:
`Atom` is a variable:
`instantiation_error`
`Atom` is neither a variable nor an atom:
`type_error(atom,Atom)`
`Atom` text cannot be read as a term:
`syntax_error(BackendDependentArgument)`

---

read_term_from_chars/3

Reads a term from a list of characters using the given read options. A period at the end of the list is optional. Valid options are those supported by the standard `read_term/3` predicate.

Compilation flags:
`static, synchronized`

Template:
`read_term_from_chars(Chars,Term,Options)`
Mode and number of proofs:
`read_term_from_chars(+list(character),-term,+list(read_option))` - `one_or_error`

Exceptions:
`Chars` contains a variable element:
`instantiation_error`
`Chars` contains an element that is not a character:
`type_error(character,Char)`
`Chars` text cannot be read as a term:
`syntax_error(BackendDependentArgument)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(read_option,Option)`

---

read_term_from_chars/4

Reads a term from a list of characters using the given read options, also returning the remaining characters. A period at the end of the term is required. Valid options are those supported by the standard `read_term/3` predicate.

Compilation flags:
`static`

Template:
`read_term_from_chars(Chars,Term,Tail,Options)`
Mode and number of proofs:
`read_term_from_chars(+list(character),-term,-list(character),+list(read_option))` - `one_or_error`

Exceptions:
`Chars` contains an element that is not a character:
`type_error(character,Char)`
`Chars` text cannot be read as a term:
`syntax_error(BackendDependentArgument)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(read_option,Option)`

---

read_from_chars/2

Reads a term from a list of characters using default read options. Shorthand for `read_term_from_chars(Chars,Term,[])`. A period at the end of the list is optional.

Compilation flags:
`static`

Template:
`read_from_chars(Chars,Term)`

Mode and number of proofs:

`read_from_chars(+list(character),-term)` - `one_or_error`

Exceptions:

`Chars` contains a variable element:

`instantiation_error`

`Chars` contains an element that is not a character:

`type_error(character,Char)`

`Chars` text cannot be read as a term:

`syntax_error(BackendDependentArgument)`

`read_term_from_codes/3`

Reads a term from a list of character codes using the given read options. A period at the end of the list is optional. Valid options are those supported by the standard `read_term/3` predicate.

Compilation flags:

`static, synchronized`

Template:

`read_term_from_codes(Codes,Term,Options)`

Mode and number of proofs:

`read_term_from_codes(+list(character_code),-term,+list(read_option))` - `one_or_error`

Exceptions:

`Codes` contains a variable element:

`instantiation_error`

`Codes` contains an element that is not a character code:

`type_error(integer,Code)`

`Codes` text cannot be read as a term:

`syntax_error(BackendDependentArgument)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(read_option,Option)`

---

read_term_from_codes/4

Reads a term from a list of character codes using the given read options, also returning the remaining character codes. A period at the end of the term is required. Valid options are those supported by the standard `read_term/3` predicate.

Compilation flags:

`static`

Template:

`read_term_from_codes(Codes,Term,Tail,Options)`

Mode and number of proofs:

`read_term_from_codes(+list(character_code),-term,-list(character_code),+list(read_option))` - `one_or_error`

Exceptions:

`Codes` contains an element that is not a character code:

`type_error(integer,Code)`

`Codes` text cannot be read as a term:

`syntax_error(BackendDependentArgument)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(read_option,Option)`

---

read_from_codes/2

Reads a term from a list of character codes using default read options. Shorthand for `read_term_from_codes(Codes,Term,[])`. A period at the end of the list is optional.

Compilation flags:

`static`

Template:

`read_from_codes(Codes,Term)`

Mode and number of proofs:

`read_from_codes(+list(character_code),-term)` - `one_or_error`

Exceptions:

`Codes` contains a variable element:

`instantiation_error`

`Codes` contains an element that is not a character code:

`type_error(integer,Code)`

`Codes` text cannot be read as a term:
`syntax_error(BackendDependentArgument)`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(read_option,Option)`

---

write_term_to_atom/3

Writes a term to an atom using the given write options. Valid options are those supported by the standard `write_term/3` predicate.

Compilation flags:
`static, synchronized`

Template:
`write_term_to_atom(Term,Atom,Options)`
Mode and number of proofs:
`write_term_to_atom(@term,-atom,+list(write_option))` - `one`

Exceptions:
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(write_option,Option)`

---

write_to_atom/2

Writes a term to an atom using default write options. Shorthand for `write_term_to_atom(Term,Atom,[])`.

Compilation flags:
`static`

Template:
`write_to_atom(Term,Atom)`
Mode and number of proofs:
`write_to_atom(@term,-atom)` - `one`

---

write_term_to_chars/3

Writes a term to a list of characters using the given write options. Shorthand for `write_term_to_chars(Term,Chars,[],Options)`. Valid options are those supported by the standard `write_term/3` predicate.

Compilation flags:
`static`

Template:
`write_term_to_chars(Term,Chars,Options)`
Mode and number of proofs:
`write_term_to_chars(@term,-list(character),+list(write_option))` - `one_or_error`

Exceptions:
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(write_option,Option)`

---

write_term_to_chars/4

Writes a term to a list of characters with the given tail using the given write options. Valid options are those supported by the standard `write_term/3` predicate.

Compilation flags:

`static, synchronized`

Template:

`write_term_to_chars(Term,Chars,Tail,Options)`

Mode and number of proofs:

`write_term_to_chars(@term,-list(character),@term,+list(write_option))` - `one_or_error`

Exceptions:

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(write_option,Option)`

---

write_to_chars/2

Writes a term to a list of characters using default write options. Shorthand for `write_term_to_chars(Term,Chars,[],[])`.

Compilation flags:

`static`

Template:

`write_to_chars(Term,Chars)`

Mode and number of proofs:

`write_to_chars(@term,-list(character))` - `one`

---

write_term_to_codes/3

Writes a term to a list of character codes using the given write options. Shorthand for `write_term_to_codes(Term,Codes,[],Options)`. Valid options are those supported by the standard `write_term/3` predicate.

Compilation flags:
`static`

Template:
`write_term_to_codes(Term,Codes,Options)`

Mode and number of proofs:
`write_term_to_codes(@term,-list(character_code),+list(write_option))` - `one_or_error`

Exceptions:
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(write_option,Option)`

---

write_term_to_codes/4

Writes a term to a list of character codes with the given tail using the given write options. Valid options are those supported by the standard `write_term/3` predicate.

Compilation flags:
`static, synchronized`

Template:
`write_term_to_codes(Term,Codes,Tail,Options)`

Mode and number of proofs:
`write_term_to_codes(@term,-list(character_code),@term,+list(write_option))` - `one_or_error`

Exceptions:
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(write_option,Option)`

---

write_to_codes/2

Writes a term to a list of character codes using default write options. Shorthand for `write_term_to_chars(Term,Codes,[],[])`.

Compilation flags:
`static`

Template:
`write_to_codes(Term,Codes)`
Mode and number of proofs:
`write_to_codes(@term,-list(character_code))` - `one`

---

format_to_atom/3

Writes a list of arguments to an atom using the given format (specified as in the de facto standard `format/2` predicate).

Compilation flags:
`static, synchronized`

Template:
`format_to_atom(Format,Arguments,Atom)`
Mode and number of proofs:
`format_to_atom(@atom,+list(term),-atom)` - `one`

---

format_to_chars/3

Writes a list of arguments to a list of characters using the given format (specified as in the de facto standard `format/2` predicate). Shorthand for `format_to_chars(Format,Arguments,Chars,[])`.

Compilation flags:
`static`

Template:
`format_to_chars(Format,Arguments,Chars)`
Mode and number of proofs:
`format_to_chars(@term,+list(term),-list(character))` - `one`

---

format_to_chars/4

Writes a term to a list of characters with the given tail using the given format (specified as in the de facto standard `format/2` predicate).

Compilation flags:

```
static, synchronized
```

Template:

```
format_to_chars(Format,Arguments,Chars,Tail)
```

Mode and number of proofs:

```
format_to_chars(@term,+list(term),-list(character),@term) - one
```

---

format_to_codes/3

Writes a list of arguments to a list of character codes using the given format (specified as in the de facto standard `format/2` predicate). Shorthand for `format_to_codes(Format,Arguments,Codes,[])`.

Compilation flags:

```
static
```

Template:

```
format_to_codes(Format,Arguments,Codes)
```

Mode and number of proofs:

```
format_to_codes(@term,+list(term),-list(character_code)) - one
```

---

format_to_codes/4

Writes a list of arguments to a list of character codes with the given tail using the given format (specified as in the de facto standard `format/2` predicate).

Compilation flags:

```
static, synchronized
```

Template:

```
format_to_codes(Format,Arguments,Codes,Tail)
```

Mode and number of proofs:

```
format_to_codes(@term,+list(term),-list(character_code),@term) - one
```

---

with_output_to/2

Calls a goal deterministically with output to the given format: `atom(Atom)`, `chars(Chars)`, `chars(Chars,Tail)`, `codes(Codes)`, or `codes(Codes,Tail)`.

Compilation flags:
`static, synchronized`

Template:
`with_output_to(Output,Goal)`
Meta-predicate template:
`with_output_to(*,0)`
Mode and number of proofs:
`with_output_to(+compound,+callable)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.317 text_normalization

object

1.317.1 default_text_normalization

Default language-neutral text normalization profile.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *text_normalization_profile_protocol*
Uses:

xml_whatwg_entities

Remarks:
(none)

Inherited public predicates:
case_conversion/3 *diacritic_fold/2* *named_entity/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

text_normalizer(Representation,Profile)

protocol

1.317.2 `spelling_normalizer_protocol`

Optional provider protocol for spelling correction composed after text normalization.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static`

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *correct_spelling_codes/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

correct_spelling_codes/3

Corrects a normalized word represented as Unicode code points using provider-defined options. Fails when no correction is available.

Compilation flags:
`static`

Template:
`correct_spelling_codes(Word,Corrected,Options)`
Mode and number of proofs:
`correct_spelling_codes(+codes,-codes,+list(compound))` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

text_normalizer_protocol

category

1.317.3 text_case_folding

Portable Unicode 17.0 default case conversion over lists of Unicode scalar values.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static`

Uses:
list
user

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *convert_case_codes/4*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

convert_case_codes/4

Applies a profile override or the default `fold`, `lower`, `upper`, or `title` case conversion.

Compilation flags:
`static`

Template:
`convert_case_codes(Mode,Codes,Profile,Converted)`
Mode and number of proofs:
`convert_case_codes(+atom,+list(integer),+object_identifier,-list(integer))` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.317.4 text_diacritics

Unicode diacritic removal and profile-defined transliteration over code lists.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static`

Extends:
`public` *text_unicode*
Uses:
list
user

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *remove_diacritics_codes/2*
  - *fold_diacritics_codes/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

remove_diacritics_codes/2

Canonically decomposes text, removes all Unicode Mark-category code points, and returns NFC text.

Compilation flags:

```
static
```

Template:

```
remove_diacritics_codes(Codes,Removed)
```

Mode and number of proofs:

```
remove_diacritics_codes(+list(integer),-list(integer)) - one_or_error
```

Exceptions:

An input element is not a Unicode scalar value:

```
domain_error(unicode_scalar_value,Code)
```

---

fold_diacritics_codes/3

Removes Unicode marks, applies profile transliterations to remaining code points, and returns NFC text.

Compilation flags:

```
static
```

Template:

```
fold_diacritics_codes(Codes,Profile,Folded)
```

Mode and number of proofs:

```
fold_diacritics_codes(+list(integer),+object_identifier,-list(integer)) -
one_or_error
```

Exceptions:

An input element is not a Unicode scalar value:

```
domain_error(unicode_scalar_value,Code)
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.317.5 `text_entities`

Portable character reference decoding rules over lists of Unicode code points.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *decode_entities_codes/4*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

decode_entities_codes/4

Decodes semicolon-terminated numeric and profile-defined named character references. The unknown-name policy is `preserve` or `error`.

Compilation flags:
`static`

Template:
`decode_entities_codes(Codes,Profile,Unknown,Decoded)`

Mode and number of proofs:
`decode_entities_codes(+list(integer),+object_identifier,+atom,-list(integer))` - `one_or_error`

Exceptions:
A numeric reference denotes a value that is not a Unicode scalar value:
`domain_error(unicode_scalar_value,Value)`
A named reference is unknown and the policy is `error`:
`domain_error(character_reference,Reference)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

text_normalization_profile_protocol, *text_normalizer(Representation,Profile)*

protocol

1.317.6 text_normalization_profile_protocol

Profile provider protocol for extending text normalization policy without replacing Unicode normalization algorithms.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *case_conversion/3*
  - *diacritic_fold/2*
  - *named_entity/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

case_conversion/3

Converts a list of Unicode code points for a profile-specific case operation. Valid operations are `fold`, `lower`, `upper`, and `title`. Fails to select the library default Unicode behavior.

Compilation flags:
`static`

Template:
`case_conversion(Operation,Codes,Converted)`

Mode and number of proofs:
`case_conversion(+atom,+codes,-codes)` - `zero_or_one`

---

diacritic_fold/2

Maps a Unicode code point without a canonical base decomposition to a profile-specific replacement. Fails when no mapping is defined.

Compilation flags:
`static`

Template:
`diacritic_fold(Code,Replacement)`

Mode and number of proofs:
`diacritic_fold(+code,-codes)` - `zero_or_one`

---

named_entity/2

Maps a case-sensitive named character reference, without the leading ampersand or trailing semicolon, to Unicode code points. Fails for an unknown name.

Compilation flags:
`static`

Template:
`named_entity(Name,Codes)`
Mode and number of proofs:
`named_entity(+atom,-codes)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

text_normalizer_protocol, *text_normalizer(Representation,Profile)*

object

1.317.7 text_normalizer(Representation,Profile)

- `Representation` - Text representation. Valid values are `atom`, `chars`, and `codes`.
- `Profile` - Object implementing the `text_normalization_profile_protocol` protocol.

Text normalizer parameterized by text representation and normalization profile.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-06

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *text_normalizer_protocol*

Imports:

`public` *options*
`public` *text_diacritics*
`public` *text_case_folding*
`public` *text_entities*
`public` *text_whitespace*

Uses:

list
type

Remarks:

(none)

Inherited public predicates:

case_fold/2 *check_option/1* *check_options/1* *clean/2* *clean/3* *decode_entities/2* *decode_entities/3* *default_option/1* *default_options/1* *fold_diacritics/2* *lower_case/2* *normalize_unicode/3* *normalize_whitespace/2* *normalize_whitespace/3* *option/2* *option/3* *remove_diacritics/2* *title_case/2* *upper_case/2* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

text_normalizer_protocol, *text_normalization_profile_protocol*, *default_text_normalization*

protocol

1.317.8 text_normalizer_protocol

Text normalization protocol for text represented as atoms, lists of characters, or lists of character codes.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *normalize_unicode/3*
  - *remove_diacritics/2*
  - *fold_diacritics/2*
  - *case_fold/2*
  - *lower_case/2*
  - *upper_case/2*
  - *title_case/2*
  - *decode_entities/2*
  - *decode_entities/3*
  - *normalize_whitespace/2*
  - *normalize_whitespace/3*
  - *clean/2*
  - *clean/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

normalize_unicode/3

Normalizes text to the specified Unicode normalization form. Valid forms are `nfc`, `nfd`, `nfkc`, and `nfkd`.

Compilation flags:

`static`

Template:

`normalize_unicode(Form,Text,Normalized)`

Mode and number of proofs:

`normalize_unicode(+atom,+text,-text)` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Profile` parameter is a variable:

`instantiation_error`

The `Profile` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Profile)`

The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:

`domain_error(text_normalization_profile,Profile)`

`Form` is a variable:

`instantiation_error`

`Form` is not an atom:

`type_error(atom,Form)`

`Form` is not a supported Unicode normalization form:

`domain_error(unicode_normalization_form,Form)`

`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

An input character code is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Code)`

remove_diacritics/2

Canonically decomposes text, removes Unicode mark characters, and returns NFC text.

Compilation flags:
`static`

Template:
`remove_diacritics(Text,Normalized)`
Mode and number of proofs:
`remove_diacritics(+text,-text)` - `one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Profile` parameter is a variable:
`instantiation_error`
The `Profile` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Profile)`
The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:
`domain_error(text_normalization_profile,Profile)`
`Text` is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but `Text` is not an atom:
`type_error(atom,Text)`
The `Representation` parameter is `chars` but `Text` is not a list of characters:
`type_error(chars,Text)`
The `Representation` parameter is `codes` but `Text` is not a list of character codes:
`type_error(codes,Text)`
An input character code is not a Unicode scalar value:
`domain_error(unicode_scalar_value,Code)`

---

fold_diacritics/2

Removes diacritics and applies profile-defined transliterations for characters without canonical decompositions.

Compilation flags:
`static`

Template:
`fold_diacritics(Text,Normalized)`
Mode and number of proofs:
`fold_diacritics(+text,-text)` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Profile` parameter is a variable:
`instantiation_error`
The `Profile` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Profile)`
The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:
`domain_error(text_normalization_profile,Profile)`
`Text` is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but `Text` is not an atom:
`type_error(atom,Text)`
The `Representation` parameter is `chars` but `Text` is not a list of characters:
`type_error(chars,Text)`
The `Representation` parameter is `codes` but `Text` is not a list of character codes:
`type_error(codes,Text)`
An input character code is not a Unicode scalar value:
`domain_error(unicode_scalar_value,Code)`

`case_fold/2`

Applies full default Unicode case folding, unless overridden by the selected profile.

Compilation flags:
`static`

Template:
`case_fold(Text,Folded)`
Mode and number of proofs:
`case_fold(+text,-text)` - `one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Profile` parameter is a variable:
`instantiation_error`
The `Profile` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Profile)`
The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:
`domain_error(text_normalization_profile,Profile)`
`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

An input character code is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Code)`

`lower_case/2`

Converts text to lowercase using the selected profile and Unicode default casing.

Compilation flags:

`static`

Template:

`lower_case(Text,Lowercase)`

Mode and number of proofs:

`lower_case(+text,-text)` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Profile` parameter is a variable:

`instantiation_error`

The `Profile` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Profile)`

The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:

`domain_error(text_normalization_profile,Profile)`

`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

An input character code is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Code)`

---

upper_case/2

Converts text to uppercase using the selected profile and Unicode default casing.

Compilation flags:
`static`

Template:
`upper_case(Text,Uppercase)`
Mode and number of proofs:
`upper_case(+text,-text)` - `one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Profile` parameter is a variable:
`instantiation_error`
The `Profile` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Profile)`
The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:
`domain_error(text_normalization_profile,Profile)`
`Text` is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but `Text` is not an atom:
`type_error(atom,Text)`
The `Representation` parameter is `chars` but `Text` is not a list of characters:
`type_error(chars,Text)`
The `Representation` parameter is `codes` but `Text` is not a list of character codes:
`type_error(codes,Text)`
An input character code is not a Unicode scalar value:
`domain_error(unicode_scalar_value,Code)`

---

title_case/2

Converts text to title case using the library simple word-boundary definition.

Compilation flags:
`static`

Template:
`title_case(Text,Titlecase)`
Mode and number of proofs:
`title_case(+text,-text)` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Profile` parameter is a variable:
`instantiation_error`
The `Profile` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Profile)`
The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:
`domain_error(text_normalization_profile,Profile)`
`Text` is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but `Text` is not an atom:
`type_error(atom,Text)`
The `Representation` parameter is `chars` but `Text` is not a list of characters:
`type_error(chars,Text)`
The `Representation` parameter is `codes` but `Text` is not a list of character codes:
`type_error(codes,Text)`
An input character code is not a Unicode scalar value:
`domain_error(unicode_scalar_value,Code)`

`decode_entities/2`

Decodes semicolon-terminated XML, numeric, and profile-defined named character references using default options.

Compilation flags:
`static`

Template:
`decode_entities(Text,Decoded)`
Mode and number of proofs:
`decode_entities(+text,-text)` - `one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Profile` parameter is a variable:
`instantiation_error`
The `Profile` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Profile)`
The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:
`domain_error(text_normalization_profile,Profile)`
`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

An input character code is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Code)`

A numeric character reference denotes a value that is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Value)`

`decode_entities/3`

Decodes semicolon-terminated character references. The recognized option is `unknown(preserve|error)`.

Compilation flags:

`static`

Template:

`decode_entities(Text,Decoded,Options)`

Mode and number of proofs:

`decode_entities(+text,-text,+list(compound))` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Profile` parameter is a variable:

`instantiation_error`

The `Profile` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Profile)`

The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:

`domain_error(text_normalization_profile,Profile)`

`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

An input character code is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Code)`

A numeric character reference denotes a value that is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Value)`

A named character reference is unknown and the `unknown(error)` option is used:

`domain_error(character_reference,Reference)`

`normalize_whitespace/2`

Normalizes whitespace using default options.

Compilation flags:

`static`

Template:

`normalize_whitespace(Text,Normalized)`

Mode and number of proofs:

`normalize_whitespace(+text,-text)` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Profile` parameter is a variable:

`instantiation_error`

The `Profile` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Profile)`

The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:

`domain_error(text_normalization_profile,Profile)`

`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

An input character code is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Code)`

`normalize_whitespace/3`

Normalizes whitespace. Recognized options are `trim(Boolean)`, `collapse(none|horizontal|all)`, `line_endings(lf|crlf|cr|preserve)`, and `controls(preserve|remove)`.

Compilation flags:

`static`

Template:

`normalize_whitespace(Text,Normalized,Options)`

Mode and number of proofs:

`normalize_whitespace(+text,-text,+list(compound))` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Profile` parameter is a variable:

`instantiation_error`

The `Profile` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Profile)`

The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:

`domain_error(text_normalization_profile,Profile)`

`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

An input character code is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Code)`

clean/2

Cleans text using the default normalization pipeline options.

Compilation flags:

`static`

Template:

`clean(Text,Cleaned)`

Mode and number of proofs:

`clean(+text,-text)` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Profile` parameter is a variable:

`instantiation_error`

The `Profile` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Profile)`

The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:

`domain_error(text_normalization_profile,Profile)`

`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

An input character code is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Code)`

A numeric character reference denotes a value that is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Value)`

clean/3

Cleans text by applying entity decoding, case conversion, diacritic handling, Unicode normalization, and whitespace normalization in that order.

Compilation flags:

`static`

Template:

`clean(Text,Cleaned,Options)`

Mode and number of proofs:

`clean(+text,-text,+list(compound))` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Profile` parameter is a variable:

`instantiation_error`

The `Profile` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Profile)`

The `Profile` parameter is an object identifier but not an object conforming to `text_normalization_profile_protocol`:

`domain_error(text_normalization_profile,Profile)`

`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

An input character code is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Code)`

A numeric character reference denotes a value that is not a Unicode scalar value:

`domain_error(unicode_scalar_value,Value)`

A named character reference is unknown and the `unknown_entities(error)` option is used:

`domain_error(character_reference,Reference)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

text_normalization_profile_protocol, *spelling_normalizer_protocol*, *text_normalizer(Representation,Profile)*

category

1.317.9 `text_unicode`

Portable Unicode 17.0 normalization over lists of Unicode scalar values.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static`

Uses:
list
user

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *normalize_unicode_codes/3*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

`normalize_unicode_codes/3`

Normalizes code points using the NFC, NFD, NFKC, or NFKD normalization form.

Compilation flags:
`static`

Template:
`normalize_unicode_codes(Form,Codes,Normalized)`
Mode and number of proofs:
`normalize_unicode_codes(+atom,+list(integer),-list(integer))` - `one_or_error`

Exceptions:
An input element is not a Unicode scalar value:
`domain_error(unicode_scalar_value,Code)`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.317.10 `text_whitespace`

Portable Unicode whitespace, line-ending, and control-character normalization rules over code lists.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static`

Uses:
string(Representation)
user

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *normalize_whitespace_codes/6*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

normalize_whitespace_codes/6

Normalizes code-list whitespace using trim, collapse, line-ending, and control-character policies.

Compilation flags:
```
static
```

Template:
```
normalize_whitespace_codes(Codes,Trim,Collapse,LineEndings,Controls,Normalized)
```

Mode and number of proofs:
```
normalize_whitespace_codes(+list(integer),+boolean,+atom,+atom,+atom,
-list(integer)) - one
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

text_normalizer(Representation,Profile), *string(Representation)*

object

1.317.11 xml_whatwg_entities

XML and WHATWG HTML semicolon-terminated named character references.

Availability:
`logtalk_load(text_normalization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-06

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *xml_whatwg_named_entity/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

xml_whatwg_named_entity/2

XML or WHATWG HTML named character reference mapping.

Compilation flags:
`static`

Template:
`xml_whatwg_named_entity(Name,Codes)`
Mode and number of proofs:
`xml_whatwg_named_entity(+atom,-list(integer))` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.318 text_vectorization

object

1.318.1 `text_vectorizer`

Binary, count, term-frequency, and TF-IDF sparse text vectorization.

Availability:
`logtalk_load(text_vectorization(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-04

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *text_vectorizer_protocol*
Imports:
`public` *options*
Uses:
list
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_vectorizer/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *learn/2* *learn/3* *learn_transform/3* *learn_transform/4* *option/2* *option/3* *transform/3* *transform_all/3* *valid_option/1* *valid_options/1* *valid_vectorizer/1* *vectorizer_options/2* *vocabulary/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

text_vectorizer_protocol

protocol

1.318.2 `text_vectorizer_protocol`

Protocol for learning and applying text vectorization models.

Availability:
`logtalk_load(text_vectorization(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/2*
  - *learn/3*
  - *transform/3*
  - *transform_all/3*

    - *learn_transform/3*
    - *learn_transform/4*
    - *check_vectorizer/1*
    - *valid_vectorizer/1*
    - *vocabulary/2*
    - *diagnostics/2*
    - *diagnostic/2*
    - *vectorizer_options/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/2

Learns a vectorizer from a non-empty corpus using default options. Documents are lists of ground features.

Compilation flags:

`static`

Template:

`learn(Corpus,Vectorizer)`

Mode and number of proofs:

`learn(+list(list),-compound)` - `one_or_error`

Exceptions:

`Corpus` is a variable or a partial list:

`instantiation_error`

`Corpus` is neither a variable nor a list:

`type_error(list,Corpus)`

`Corpus` is empty:

`domain_error(non_empty_corpus,Corpus)`

An element `Document` of the list `Corpus` is a variable or a partial list:

`instantiation_error`

An element `Document` of the list `Corpus` is neither a variable nor a list:

`type_error(list,Document)`

An element `Feature` of a document is not a ground term:

`instantiation_error`

No corpus feature satisfies the document-frequency filters:

`domain_error(non_empty_vocabulary,Corpus)`

learn/3

Learns a vectorizer from a non-empty corpus using the given options. Documents are lists of ground features.

Compilation flags:

`static`

Template:

`learn(Corpus,Vectorizer,Options)`

Mode and number of proofs:

`learn(+list(list),-compound,+list(compound))` - `one_or_error`

Exceptions:

`Corpus` is a variable or a partial list:

`instantiation_error`

`Corpus` is neither a variable nor a list:

`type_error(list,Corpus)`

`Corpus` is empty:

`domain_error(non_empty_corpus,Corpus)`

An element `Document` of the list `Corpus` is a variable or a partial list:

`instantiation_error`

An element `Document` of the list `Corpus` is neither a variable nor a list:

`type_error(list,Document)`

An element `Feature` of a document is not a ground term:

`instantiation_error`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

The `maximum_document_frequency(Maximum)` option value is greater than the corpus document count:

`domain_error(option,maximum_document_frequency(Maximum))`

The `minimum_document_frequency(Minimum)` option value is greater than the effective maximum document frequency:

`domain_error(option,minimum_document_frequency(Minimum))`

No corpus feature satisfies the document-frequency filters:

`domain_error(non_empty_vocabulary,Corpus)`

transform/3

Transforms a document into a sparse list of `Feature-Weight` pairs in standard term order. Out-of-vocabulary features are ignored.

Compilation flags:
`static`

Template:
`transform(Vectorizer,Document,Vector)`
Mode and number of proofs:
`transform(+compound,+list,-list(pair))` - `one_or_error`

Exceptions:
`Vectorizer` is a variable:
`instantiation_error`
`Vectorizer` is neither a variable nor a valid text vectorizer:
`domain_error(text_vectorizer,Vectorizer)`
`Document` is a variable or a partial list:
`instantiation_error`
`Document` is neither a variable nor a list:
`type_error(list,Document)`
An element `Feature` of the list `Document` is not a ground term:
`instantiation_error`

---

transform_all/3

Transforms all corpus documents into sparse vectors, preserving document order.

Compilation flags:
`static`

Template:
`transform_all(Vectorizer,Corpus,Vectors)`
Mode and number of proofs:
`transform_all(+compound,+list(list),-list(list(pair)))` - `one_or_error`

Exceptions:
`Vectorizer` is a variable:
`instantiation_error`
`Vectorizer` is neither a variable nor a valid text vectorizer:
`domain_error(text_vectorizer,Vectorizer)`
`Corpus` is a variable or a partial list:
`instantiation_error`
`Corpus` is neither a variable nor a list:
`type_error(list,Corpus)`
An element `Document` of the list `Corpus` is a variable or a partial list:

instantiation_error

An element Document of the list Corpus is neither a variable nor a list:

type_error(list,Document)

An element Feature of a document is not a ground term:

instantiation_error

---

learn_transform/3

Learns a vectorizer using default options and transforms the training corpus.

Compilation flags:

static

Template:

learn_transform(Corpus,Vectorizer,Vectors)

Mode and number of proofs:

learn_transform(+list(list),-compound,-list(list(pair)))-one_or_error

Exceptions:

Corpus is a variable or a partial list:

instantiation_error

Corpus is neither a variable nor a list:

type_error(list,Corpus)

Corpus is empty:

domain_error(non_empty_corpus,Corpus)

An element Document of the list Corpus is a variable or a partial list:

instantiation_error

An element Document of the list Corpus is neither a variable nor a list:

type_error(list,Document)

An element Feature of a document is not a ground term:

instantiation_error

No corpus feature satisfies the document-frequency filters:

domain_error(non_empty_vocabulary,Corpus)

---

learn_transform/4

Learns a vectorizer using the given options and transforms the training corpus.

Compilation flags:

static

Template:

learn_transform(Corpus,Vectorizer,Vectors,Options)

Mode and number of proofs:

`learn_transform(+list(list),-compound,-list(list(pair)),+list(compound))` - `one_or_error`

Exceptions:

`Corpus` is a variable or a partial list:
`instantiation_error`
`Corpus` is neither a variable nor a list:
`type_error(list,Corpus)`
`Corpus` is empty:
`domain_error(non_empty_corpus,Corpus)`
An element `Document` of the list `Corpus` is a variable or a partial list:
`instantiation_error`
An element `Document` of the list `Corpus` is neither a variable nor a list:
`type_error(list,Document)`
An element `Feature` of a document is not a ground term:
`instantiation_error`
`Options` is a variable:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`
The `maximum_document_frequency(Maximum)` option value is greater than the corpus document count:
`domain_error(option,maximum_document_frequency(Maximum))`
The `minimum_document_frequency(Minimum)` option value is greater than the effective maximum document frequency:
`domain_error(option,minimum_document_frequency(Minimum))`
No corpus feature satisfies the document-frequency filters:
`domain_error(non_empty_vocabulary,Corpus)`

`check_vectorizer/1`

Checks that a term is a structurally valid learned text vectorizer.

Compilation flags:
`static`

Template:
`check_vectorizer(Vectorizer)`
Mode and number of proofs:
`check_vectorizer(@compound)` - `one_or_error`

Exceptions:

`Vectorizer` is a variable:
`instantiation_error`
`Vectorizer` is neither a variable nor a valid text vectorizer:
`domain_error(text_vectorizer,Vectorizer)`

`valid_vectorizer/1`

True when a term is a structurally valid learned text vectorizer.

Compilation flags:
`static`

Template:
`valid_vectorizer(Vectorizer)`
Mode and number of proofs:
`valid_vectorizer(@compound)` - `zero_or_one`

`vocabulary/2`

Returns the learned vocabulary in standard term order.

Compilation flags:
`static`

Template:
`vocabulary(Vectorizer,Vocabulary)`
Mode and number of proofs:
`vocabulary(+compound,-list)` - `one_or_error`

Exceptions:
`Vectorizer` is a variable:
`instantiation_error`
`Vectorizer` is neither a variable nor a valid text vectorizer:
`domain_error(text_vectorizer,Vectorizer)`

---

diagnostics/2

Returns the diagnostics and metadata stored in a learned vectorizer.

Compilation flags:
`static`

Template:
`diagnostics(Vectorizer,Diagnostics)`
Mode and number of proofs:
`diagnostics(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Vectorizer` is a variable:
`instantiation_error`
`Vectorizer` is neither a variable nor a valid text vectorizer:
`domain_error(text_vectorizer,Vectorizer)`

---

diagnostic/2

Tests or enumerates individual diagnostics terms stored in a learned vectorizer.

Compilation flags:
`static`

Template:
`diagnostic(Vectorizer,Diagnostic)`
Mode and number of proofs:
`diagnostic(+compound,?compound)` - `zero_or_more`

Exceptions:
`Vectorizer` is a variable:
`instantiation_error`
`Vectorizer` is neither a variable nor a valid text vectorizer:
`domain_error(text_vectorizer,Vectorizer)`

---

vectorizer_options/2

Returns the effective training options stored in a learned vectorizer.

Compilation flags:
`static`

Template:
`vectorizer_options(Vectorizer,Options)`

Mode and number of proofs:
`vectorizer_options(+compound,-list(compound))` - `one_or_error`

Exceptions:
`Vectorizer` is a variable:
`instantiation_error`
`Vectorizer` is neither a variable nor a valid text vectorizer:
`domain_error(text_vectorizer,Vectorizer)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

text_vectorizer

1.319 tfidf_recommender

object

1.319.1 `tfidf_recommender`

Content-based recommender using TF-IDF item vectors, positive-feedback centroid profiles, and cosine relevance scores.

Availability:
`logtalk_load(tfidf_recommender(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-06

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *recommender_common*
`public` *item_content_dataset_validation*
`public` *similarity_metric_common*
Uses:

cosine_similarity
list
text_vectorizer
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_recommender/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_recommender/1* *recommend/4* *recommender_options/2* *score/4* *valid_option/1* *valid_options/1* *valid_recommender/1*

- *Public predicates*
  - *score_all/4*
  - *score_content/4*
  - *update_ratings/3*
  - *remove_ratings/3*
  - *extend_catalog/3*
  - *replace_content/3*
- *Protected predicates*
- *Private predicates*
  - *check_content_kind/3*
  - *check_catalog_replacements/2*
  - *check_catalog_additions/2*
  - *rebuild_catalog_model/5*
  - *rating_removal_records/3*
  - *check_rating_updates/3*
  - *rebuild_feedback_model/7*
  - *catalog_score/5*
  - *score_catalog_items/5*
  - *check_finite_ratings/1*
  - *build_vectors/5*
  - *profile_weights/3*
- *Operators*

Public predicates

score_all/4

Scores catalog items in input order, preserving duplicates and validating the model once. An empty batch validates the model and user.

Compilation flags:
`static`

Template:
`score_all(Recommender,User,Items,Scores)`

Mode and number of proofs:

```
score_all(+compound,+atomic,+list(atomic),-list(pair)) - one_or_error
```

Exceptions:

A required argument or identifier is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

Items is not a proper list:

```
type_error(list,Items)
```

An item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

---

score_content/4

Scores supplied content of the trained representation without changing the model. Features use the fitted vocabulary and weighting; vectors use the stored normalization.

Compilation flags:

```
static
```

Template:

```
score_content(Recommender,User,Content,Score)
```

Mode and number of proofs:

```
score_content(+compound,+atomic,+compound,-number) - one_or_error
```

Exceptions:

A required argument, feature, entry, or weight is a variable:

```
instantiation_error
```

The model is invalid:

```
domain_error(recommender,Recommender)
```

User is not atomic:

```
type_error(atomic,User)
```

A content descriptor is invalid:

```
domain_error(item_content,Content)
```

The descriptor differs from the trained representation:

```
domain_error(content_representation,Content)
```

A content list is not a proper list:

```
type_error(list,List)
```

A vector entry is not a pair:

```
type_error(pair,Entry)
```

A vector repeats a feature key:

```
domain_error(duplicate_feature,Feature)
```

A vector weight is not numeric:

```
type_error(number,Weight)
```

A vector weight is negative or nonfinite:
`domain_error(non_negative_finite_weight,Weight)`

---

update_ratings/3

Returns a model with inserted or replaced ratings and rebuilt profiles, accepting new users but requiring catalog items. An empty update returns the validated original model.

Compilation flags:
`static`

Template:
`update_ratings(Recommender,Ratings,UpdatedRecommender)`

Mode and number of proofs:
`update_ratings(+compound,+list(compound),-compound)` - `one_or_error`

Exceptions:
A required argument, record, identifier, or value is a variable:
`instantiation_error`
The model is invalid:
`domain_error(recommender,Recommender)`
Ratings is not a proper list:
`type_error(list,Ratings)`
An entry is not a rating record:
`type_error(rating,Entry)`
A user or item identifier is not atomic:
`type_error(atomic,Identifier)`
A rating value is not numeric:
`type_error(number,Rating)`
A rating value is nonfinite:
`domain_error(finite_rating,Rating)`
A rating value is outside the stored scale:
`domain_error(rating_scale(Min,Max),Rating)`
An item is absent from the catalog:
`domain_error(catalog_item,Item)`
A user-item pair occurs more than once in the updates:
`domain_error(duplicate_rating,User-Item)`
A selected rating is not positive in rating-weighted mode:
`domain_error(positive_rating_weight,Rating)`

---

remove_ratings/3

Returns a model with requested ratings removed and profiles rebuilt. Missing and repeated pairs are accepted; no matching ratings returns the validated original model.

Compilation flags:

`static`

Template:

`remove_ratings(Recommender,UserItemPairs,UpdatedRecommender)`

Mode and number of proofs:

`remove_ratings(+compound,+list(pair),-compound)` - `one_or_error`

Exceptions:

A required argument, entry, or identifier is a variable:

`instantiation_error`

The model is invalid:

`domain_error(recommender,Recommender)`

UserItemPairs is not a proper list:

`type_error(list,UserItemPairs)`

An entry is not a pair:

`type_error(pair,Entry)`

A user or item identifier is not atomic:

`type_error(atomic,Identifier)`

No ratings would remain:

`domain_error(non_empty_ratings,[])`

A selected rating is not positive in rating-weighted mode:

`domain_error(positive_rating_weight,Rating)`

---

extend_catalog/3

Returns a model with new catalog items, refitting the full feature corpus and rebuilding profiles. An empty extension returns the validated original model.

Compilation flags:

`static`

Template:

`extend_catalog(Recommender,ItemContents,UpdatedRecommender)`

Mode and number of proofs:

`extend_catalog(+compound,+list(pair),-compound)` - `one_or_error`

Exceptions:

A required argument, entry, identifier, feature, or weight is a variable:

`instantiation_error`

The model is invalid:

`domain_error(recommender,Recommender)`

A supplied list is not a proper list:
`type_error(list,List)`
A content declaration or vector entry is not a pair:
`type_error(pair,Entry)`
An item identifier is not atomic:
`type_error(atomic,Item)`
An identifier already belongs to the catalog:
`domain_error(new_catalog_item,Item)`
An identifier occurs more than once in the extension:
`domain_error(duplicate_item,Item)`
A content descriptor is invalid:
`domain_error(item_content,Content)`
Content representations are mixed or differ from the catalog:
`domain_error(content_representation,Content)`
A vector repeats a feature key:
`domain_error(duplicate_feature,Feature)`
A vector weight is not numeric:
`type_error(number,Weight)`
A vector weight is negative or nonfinite:
`domain_error(non_negative_finite_weight,Weight)`
Feature fitting produces no vocabulary:
`domain_error(non_empty_vocabulary,Corpus)`
A vectorizer frequency bound is inconsistent with the corpus:
`domain_error(option,Option)`
A selected rating is not positive in rating-weighted mode:
`domain_error(positive_rating_weight,Rating)`

---

replace_content/3

Returns a model with replaced catalog content, refitting the full feature corpus and rebuilding profiles. An empty replacement returns the validated original model.

Compilation flags:
`static`

Template:
`replace_content(Recommender,ItemContents,UpdatedRecommender)`
Mode and number of proofs:
`replace_content(+compound,+list(pair),-compound)` - `one_or_error`

Exceptions:
A required argument, entry, identifier, feature, or weight is a variable:
`instantiation_error`
The model is invalid:
`domain_error(recommender,Recommender)`
A supplied list is not a proper list:
`type_error(list,List)`

A content declaration or vector entry is not a pair:
`type_error(pair,Entry)`
An item identifier is not atomic:
`type_error(atomic,Item)`
An item is absent from the catalog:
`domain_error(catalog_item,Item)`
An identifier occurs more than once in the replacements:
`domain_error(duplicate_item,Item)`
A content descriptor is invalid:
`domain_error(item_content,Content)`
Content representations are mixed or differ from the catalog:
`domain_error(content_representation,Content)`
A vector repeats a feature key:
`domain_error(duplicate_feature,Feature)`
A vector weight is not numeric:
`type_error(number,Weight)`
A vector weight is negative or nonfinite:
`domain_error(non_negative_finite_weight,Weight)`
Feature fitting produces no vocabulary:
`domain_error(non_empty_vocabulary,Corpus)`
A vectorizer frequency bound is inconsistent with the corpus:
`domain_error(option,Option)`
A selected rating is not positive in rating-weighted mode:
`domain_error(positive_rating_weight,Rating)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`check_content_kind/3`

Checks that supplied content matches the stored representation.

Compilation flags:
`static`

Template:
`check_content_kind(Kind,Content,Diagnostics)`
Mode and number of proofs:
`check_content_kind(+atom,+compound,+list(compound))` - `one_or_error`

Exceptions:
The descriptor differs from the trained representation:
`domain_error(content_representation,Content)`

---

check_catalog_replacements/2

Checks that replacement declarations have atomic catalog identifiers.

Compilation flags:

```
static
```

Template:

```
check_catalog_replacements(ItemContents,Contents)
```

Mode and number of proofs:

```
check_catalog_replacements(+list(pair),+list(pair)) - one_or_error
```

Exceptions:

An entry or identifier is a variable:

```
instantiation_error
```

An entry is not a pair:

```
type_error(pair,Entry)
```

An identifier is not atomic:

```
type_error(atomic,Item)
```

An identifier is absent from the catalog:

```
domain_error(catalog_item,Item)
```

---

check_catalog_additions/2

Checks that content declarations have atomic identifiers absent from the catalog.

Compilation flags:

```
static
```

Template:

```
check_catalog_additions(ItemContents,Contents)
```

Mode and number of proofs:

```
check_catalog_additions(+list(pair),+list(pair)) - one_or_error
```

Exceptions:

An entry or identifier is a variable:

```
instantiation_error
```

An entry is not a pair:

```
type_error(pair,Entry)
```

An identifier is not atomic:

```
type_error(atomic,Item)
```

An identifier already belongs to the catalog:

```
domain_error(new_catalog_item,Item)
```

---

rebuild_catalog_model/5

Rebuilds the full catalog fitted state, profiles, and diagnostics using the stored options.

Compilation flags:

```
static
```

Template:

```
rebuild_catalog_model(Ratings,Contents,Scale,Diagnostics,UpdatedRecommender)
```

Mode and number of proofs:

```
rebuild_catalog_model(+list(compound),+list(pair),+term,+list(compound),
-compound) - one_or_error
```

Exceptions:

Feature fitting produces no vocabulary:

```
domain_error(non_empty_vocabulary,Corpus)
```

A vectorizer frequency bound is inconsistent with the corpus:

```
domain_error(option,Option)
```

Vectorizer options are supplied for preweighted content:

```
domain_error(option,vectorizer_options(Options))
```

A selected rating is not positive in rating-weighted mode:

```
domain_error(positive_rating_weight,Rating)
```

---

rating_removal_records/3

Validates user-item pairs and collects matching stored ratings, ignoring missing pairs.

Compilation flags:

```
static
```

Template:

```
rating_removal_records(UserItemPairs,Ratings,Removals)
```

Mode and number of proofs:

```
rating_removal_records(+list(pair),+list(compound),-list(compound)) - one_or_error
```

Exceptions:

An entry or identifier is a variable:

```
instantiation_error
```

An entry is not a pair:

```
type_error(pair,Entry)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

---

check_rating_updates/3

Checks rating records against the catalog and stored rating scale.

Compilation flags:

```
static
```

Template:

```
check_rating_updates(Ratings,Vectors,Scale)
```

Mode and number of proofs:

```
check_rating_updates(+list(compound),+list(pair),+term) - one_or_error
```

Exceptions:

A record, identifier, or value is a variable:

```
instantiation_error
```

An entry is not a rating record:

```
type_error(rating,Entry)
```

A user or item identifier is not atomic:

```
type_error(atomic,Identifier)
```

A rating value is not numeric:

```
type_error(number,Rating)
```

A rating value is nonfinite:

```
domain_error(finite_rating,Rating)
```

A rating value is outside the stored scale:

```
domain_error(rating_scale(Min,Max),Rating)
```

An item is absent from the catalog:

```
domain_error(catalog_item,Item)
```

---

rebuild_feedback_model/7

Rebuilds profiles and diagnostics while retaining the fitted catalog state.

Compilation flags:

```
static
```

Template:

```
rebuild_feedback_model(Ratings,Contents,Vectors,Vectorizer,Scale,Diagnostics,
UpdatedRecommender)
```

Mode and number of proofs:

```
rebuild_feedback_model(+list(compound),+list(pair),+list(pair),+term,+term,
+list(compound),-compound) - one_or_error
```

Exceptions:

A selected rating is not positive in rating-weighted mode:

```
domain_error(positive_rating_weight,Rating)
```

---

catalog_score/5

Scores a catalog identifier using validated model data.

Compilation flags:
`static`

Template:
`catalog_score(Vectors,Profiles,User,Item,Score)`
Mode and number of proofs:
`catalog_score(+list(pair),+list(pair),+atomic,+atomic,-number)` - `one_or_error`

Exceptions:
An item is absent from the catalog:
`domain_error(catalog_item,Item)`

---

score_catalog_items/5

Validates and scores catalog identifiers in input order using validated model data.

Compilation flags:
`static`

Template:
`score_catalog_items(Vectors,Profiles,User,Items,Scores)`
Mode and number of proofs:
`score_catalog_items(+list(pair),+list(pair),+atomic,+list(atomic),-list(pair))` - `one_or_error`

Exceptions:
An identifier is a variable:
`instantiation_error`
An identifier is not atomic:
`type_error(atomic,Item)`
An item is absent from the catalog:
`domain_error(catalog_item,Item)`

---

check_finite_ratings/1

Checks that rating values are finite.

Compilation flags:
`static`

Template:

```
check_finite_ratings(Ratings)
```

Mode and number of proofs:

```
check_finite_ratings(+list(compound)) - one_or_error
```

Exceptions:

A rating is not finite:

```
domain_error(finite_rating,Rating)
```

---

build_vectors/5

Fits catalog feature vectors or normalizes supplied vectors using the effective options.

Compilation flags:

```
static
```

Template:

```
build_vectors(Kind,Contents,Options,Vectorizer,Vectors)
```

Mode and number of proofs:

```
build_vectors(+atom,+list(pair),+list(compound),-term,-list(pair)) - one_or_error
```

Exceptions:

Feature fitting produces no vocabulary:

```
domain_error(non_empty_vocabulary,Corpus)
```

A vectorizer frequency bound is inconsistent with the corpus:

```
domain_error(option,Option)
```

Vectorizer options are supplied for preweighted content:

```
domain_error(option,vectorizer_options(Options))
```

---

profile_weights/3

Assigns uniform or strictly positive raw-rating weights to selected item vectors.

Compilation flags:

```
static
```

Template:

```
profile_weights(Selected,Weighting,Weighted)
```

Mode and number of proofs:

```
profile_weights(+list(pair),+atom,-list(pair)) - one_or_error
```

Exceptions:

A selected rating is not positive in rating-weighted mode:

```
domain_error(positive_rating_weight,Rating)
```

Operators

(none)

See also

recommender_protocol, *item_content_dataset_protocol*, *text_vectorizer*, *cosine_similarity*

1.320 theta_forecasting

object

1.320.1 `theta_forecasting`

Standard Theta forecasting combining simple exponential smoothing and linear trend extrapolation.

Availability:
`logtalk_load(theta_forecasting(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-14

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *forecaster_common*
`public` *exponential_smoothing_common*

Uses:
format
list
nelder_mead(Problem)
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_forecaster/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *forecast/3* *forecaster_options/2* *learn/2* *learn/3* *option/2* *option/3* *print_forecaster/1* *valid_forecaster/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *fitted_values/2*
  - *fitted_values/3*

- *Protected predicates*
- *Private predicates*
  - *prepare_learning/4*
  - *prepare_seasonality/8*
  - *positive_series/1*
  - *training_totals/6*
  - *score_seasonal/9*
  - *anchor_series/2*
  - *fit_level/13*
  - *theta_correction/3*
  - *correction_steps/4*
  - *residual_totals/3*
  - *valid_model/1*
  - *valid_seasonality/4*
  - *valid_fitting/2*
- *Operators*

Public predicates

fitted_values/2

Returns aligned SES training fits on the original scale using default options. Missing targets and the first known initialization anchor have independent unbound placeholders. These are not prefix-wise Theta forecasts.

Compilation flags:

```
static
```

Template:

```
fitted_values(Dataset,Values)
```

Mode and number of proofs:

```
fitted_values(+object_identifier,-list) - one_or_error
```

Exceptions:

A length or frequency is unbound:

```
instantiation_error
```

An observation is not numeric:

```
type_error(number,Observation)
```

The declared length is not an integer:

```
type_error(integer,DeclaredLength)
```

The declared length is not positive:

```
domain_error(positive_integer,DeclaredLength)
```

The dataset has no observations:

```
domain_error(non_empty_series,Dataset)
```

Indices are not gap-free and 1-based:

```
domain_error(series_index_sequence,Dataset)
```

Declared and observed lengths differ:

```
consistency_error(series_length,DeclaredLength,ObservedLength)
```

There are fewer than two observations:

```
domain_error(series_length,Dataset)
```

There are fewer than two known observations:

```
domain_error(insufficient_known_observations,Dataset)
```

The dataset frequency is not an integer:

```
type_error(integer,Frequency)
```

The dataset frequency is not positive:

```
domain_error(positive_integer,Frequency)
```

Seasonal decomposition has fewer than two cycles:

```
domain_error(series_length,Series)
```

A seasonal phase has no valid centered-window deviation:

```
domain_error(insufficient_seasonal_phase_observations,Phase)
```

Multiplicative factors are not positive:

```
domain_error(positive_number,Value)
```

The residual count is inconsistent:

```
domain_error(residual_count,Residuals)
```

Fitting or fit restoration arithmetic fails:

```
evaluation_error(Error)
```

---

fitted_values/3

Returns aligned SES training fits on the original scale using the learning options. Missing targets and the first known initialization anchor have independent unbound placeholders. Parameters and seasonal factors are fitted once using the full sample, not individual prefixes.

Compilation flags:

```
static
```

Template:

```
fitted_values(Dataset,Values,Options)
```

Mode and number of proofs:

```
fitted_values(+object_identifier,-list,+list(compound))-one_or_error
```

Exceptions:

An option, length, or strict-policy observation is unbound:

```
instantiation_error
```

Options are not a list:

```
type_error(list,Options)
```

An option is not a compound term:

```
type_error(compound,Option)
```

An option is invalid:

```
domain_error(option,Option)
```

An observation is not numeric:

```
type_error(number,Observation)
```

The declared length is not an integer:

```
type_error(integer,DeclaredLength)
```

The declared length is not positive:

```
domain_error(positive_integer,DeclaredLength)
```

The dataset has no observations:

```
domain_error(non_empty_series,Dataset)
```

Indices are not gap-free and 1-based:

```
domain_error(series_index_sequence,Dataset)
```

Declared and observed lengths differ:

```
consistency_error(series_length,DeclaredLength,ObservedLength)
```

There are fewer than two observations:

```
domain_error(series_length,Dataset)
```

There are fewer than two known observations:

```
domain_error(insufficient_known_observations,Dataset)
```

The frequency is not an integer:

```
type_error(integer,Frequency)
```

The frequency is not positive:

```
domain_error(positive_integer,Frequency)
```

Forced seasonality needs a frequency greater than one:

```
domain_error(seasonal_frequency,Frequency)
```

Forced seasonality has no dataset frequency:

```
domain_error(seasonal_frequency,Dataset)
```

An explicit frequency accompanies no seasonality:

```
domain_error(theta_forecasting_option,Option)
```

The series has fewer than two seasonal cycles:

```
domain_error(series_length,Series)
```

A seasonal phase has no valid centered-window deviation:

```
domain_error(insufficient_seasonal_phase_observations,Phase)
```

Multiplicative data or factors are not positive:

```
domain_error(positive_number,Value)
```

The residual count is inconsistent:

```
domain_error(residual_count,Residuals)
```

Fitting or fit restoration arithmetic fails:

```
evaluation_error(Error)
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

prepare_learning/4

Collects and fits once, returning transient adjusted observations and residuals alongside the forecaster.

Compilation flags:

```
static
```

Template:

```
prepare_learning(Dataset,Forecaster,FitData,Options)
```

Mode and number of proofs:

`prepare_learning(+object_identifier,-compound,-compound,+list(compound))` - one_or_error

Exceptions:

An option, length, or strict-policy observation is unbound:
`instantiation_error`
Options are not a list:
`type_error(list,Options)`
An option is not a compound term:
`type_error(compound,Option)`
An option is invalid:
`domain_error(option,Option)`
An observation is not numeric:
`type_error(number,Observation)`
The declared length is not an integer:
`type_error(integer,DeclaredLength)`
The declared length is not positive:
`domain_error(positive_integer,DeclaredLength)`
The dataset has no observations:
`domain_error(non_empty_series,Dataset)`
Indices are not gap-free and 1-based:
`domain_error(series_index_sequence,Dataset)`
Declared and observed lengths differ:
`consistency_error(series_length,DeclaredLength,ObservedLength)`
There are fewer than two observations:
`domain_error(series_length,Dataset)`
There are fewer than two known observations:
`domain_error(insufficient_known_observations,Dataset)`
The frequency is not an integer:
`type_error(integer,Frequency)`
The frequency is not positive:
`domain_error(positive_integer,Frequency)`
Forced seasonality needs a frequency greater than one:
`domain_error(seasonal_frequency,Frequency)`
Forced seasonality has no dataset frequency:
`domain_error(seasonal_frequency,Dataset)`
An explicit frequency accompanies no seasonality:
`domain_error(theta_forecasting_option,Option)`
The series has fewer than two seasonal cycles:
`domain_error(series_length,Series)`
A seasonal phase has no valid centered-window deviation:
`domain_error(insufficient_seasonal_phase_observations,Phase)`
Multiplicative data or factors are not positive:
`domain_error(positive_number,Value)`
Fitting arithmetic fails:
`evaluation_error(Error)`

prepare_seasonality/8

Resolves the requested seasonal adjustment and dataset frequency.

Compilation flags:

```
static
```

Template:

```
prepare_seasonality(Dataset,Series,UserOptions,Options,Adjusted,Seasonality,Mode,
Frequency)
```

Mode and number of proofs:

```
prepare_seasonality(+object_identifier,+list,+list,+list,-list,-term,-atom,
-integer) - one_or_error
```

Exceptions:

The frequency is unbound:

```
instantiation_error
```

The frequency is not an integer:

```
type_error(integer,Frequency)
```

The frequency is not positive:

```
domain_error(positive_integer,Frequency)
```

Forced seasonality has no dataset frequency:

```
domain_error(seasonal_frequency,Dataset)
```

Forced seasonality has frequency one:

```
domain_error(seasonal_frequency,Frequency)
```

A frequency option is supplied with no seasonality:

```
domain_error(theta_forecasting_option,Option)
```

The series has fewer than two cycles:

```
domain_error(series_length,Series)
```

A seasonal phase has no valid centered-window deviation:

```
domain_error(insufficient_seasonal_phase_observations,Phase)
```

Multiplicative data or factors are not positive:

```
domain_error(positive_number,Value)
```

Seasonal arithmetic fails:

```
evaluation_error(Error)
```

---

positive_series/1

Checks whether all known numeric observations are strictly positive, ignoring unbound observations.

Compilation flags:

```
static
```

Template:

```
positive_series(Series)
```

Mode and number of proofs:

```
positive_series(+list) - zero_or_one
```

Exceptions:

Numeric comparison fails:

```
evaluation_error(Error)
```

---

training_totals/6

Scores SES training fits after restoring the original seasonal scale and returns the same compact scoring errors.

Compilation flags:

```
static
```

Template:

```
training_totals(Series,Adjusted,Seasonality,Errors,Totals,OriginalErrors)
```

Mode and number of proofs:

```
training_totals(+list,+list,+term,+compound,-compound,-list(number)) -
one_or_error
```

Exceptions:

Training-error arithmetic fails:

```
evaluation_error(Error)
```

---

score_seasonal/9

Scores known targets on their original scale, advancing phases but not residuals at gaps.

Compilation flags:

```
static
```

Template:

```
score_seasonal(Actuals,Adjusted,Residuals,Method,Factors,Cycle,Totals0,Totals,
OriginalErrors)
```

Mode and number of proofs:

```
score_seasonal(+list,+list,+list(number),+atom,+list(number),+list(number),
+compound,-compound,-list(number)) - one_or_error
```

Exceptions:

Fit restoration or error accumulation fails:

```
evaluation_error(Error)
```

---

anchor_series/2

Removes only leading missing positions before fitting the SES initialization anchor.

Compilation flags:

```
static
```

Template:

```
anchor_series(Series,FitSeries)
```

Mode and number of proofs:

```
anchor_series(+list,-list) - one
```

---

fit_level/13

Fits the SES level using the existing bounded optimization problem when needed.

Compilation flags:

```
static
```

Template:

```
fit_level(Series,RequestedAlpha,Initialization,OptimizerOptions,Alpha,
InitialLevel,Level,SSE,Errors,Fitting,Convergence,Iterations,Evaluations)
```

Mode and number of proofs:

```
fit_level(+compound,+term,+atom,+list,-number,-number,-number,-compound,
-atom,-atom,-integer,-integer) - one_or_error
```

Exceptions:

Optimizer options are invalid:

```
domain_error(option,Option)
```

Fitting arithmetic fails:

```
evaluation_error(Error)
```

---

theta_correction/3

Computes the finite-sample correction, including the zero-alpha limit.

Compilation flags:

```
static
```

Template:

```
theta_correction(Length,Alpha,Correction)
```

Mode and number of proofs:

```
theta_correction(+positive_integer,+number,-number) - one_or_error
```

Exceptions:

Correction arithmetic fails:
`evaluation_error(Error)`

---

correction_steps/4

Evaluates the geometric correction through a stable recurrence.

Compilation flags:
`static`

Template:
`correction_steps(Remaining,Weight,Previous,Correction)`
Mode and number of proofs:
`correction_steps(+non_negative_integer,+number,+number,-number)` - `one_or_error`

Exceptions:
Correction arithmetic fails:
`evaluation_error(Error)`

---

residual_totals/3

Accumulates training residual error totals.

Compilation flags:
`static`

Template:
`residual_totals(Residuals,Totals0,Totals)`
Mode and number of proofs:
`residual_totals(+list(number),+compound,-compound)` - `one_or_error`

Exceptions:
Error accumulation fails:
`evaluation_error(Error)`

---

valid_model/1

Validates ground state and mandatory diagnostics.

Compilation flags:
`static`

Template:

```
valid_model(Forecaster)
```

Mode and number of proofs:

```
valid_model(+compound) - zero_or_one
```

Exceptions:

Validation arithmetic fails:

```
evaluation_error(Error)
```

---

valid_seasonality/4

Validates retained seasonal factors and next forecast phase.

Compilation flags:

```
static
```

Template:

```
valid_seasonality(Seasonality,Length,Mode,Frequency)
```

Mode and number of proofs:

```
valid_seasonality(+term,+positive_integer,?atom,?positive_integer) - zero_or_one
```

Exceptions:

Phase arithmetic fails:

```
evaluation_error(Error)
```

---

valid_fitting/2

Validates fitting provenance and optimizer counters.

Compilation flags:

```
static
```

Template:

```
valid_fitting(Initialization,Diagnostics)
```

Mode and number of proofs:

```
valid_fitting(+atom,+list(compound)) - zero_or_one
```

Exceptions:

Counter comparison fails:

```
evaluation_error(Error)
```

Operators

(none)

1.321 thurstone_mosteller_ranker

object

1.321.1 `thurstone_mosteller_ranker`

Thurstone-Mosteller Case V pairwise preference ranker. Learns one real-valued latent utility per item from a dataset object implementing the `pairwise_ranking_dataset_protocol` protocol by fitting continuity-corrected empirical paired-comparison probabilities with a deterministic weighted least-squares normal model, and returns a self-describing ranker term with diagnostics that can be used for ranking and export.

Availability:
`logtalk_load(thurstone_mosteller_ranker(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-08-11

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *ranking_dataset_common*
`public` *score_ranker_model_common*

Uses:
avltree
list
numberlist
univariate_distributions(Random)

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2*

validate_multiplayer_dataset/1 *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pairwise_ranking_dataset_protocol, *ranker_protocol*, *bradley_terry_ranker*, *regularized_bradley_terry_ranker*

1.322 time_scales

object

1.322.1 `time_scales`

Time scale conversions for UTC, TAI, TT, UT1, TDB, GPS, GST, TCG, and TCB using bundled and optional override data.

Availability:

```
logtalk_load(time_scales(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-04-07

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *time_scales_protocol*

Uses:

date
time_scales_data

Remarks:

- Supported UTC range: From 1972-01-01T00:00:00Z onwards.
- Time representation: Instants are represented as `instant(Scale,Seconds,fraction(Numerator,Denominator))` with normalized fractional part in the `[0,1[` interval.

Inherited public predicates:

check_conversion/3 *check_convert/4* *check_instant/1* *check_offset/3* *clear_dut1_override/0* *clear_leap_seconds_override/0* *convert/4* *dut1_entries/1* *dut1_source/1* *instant_to_utc_date_time/2* *leap_second_date/2* *leap_seconds_entries/1* *leap_seconds_source/1* *load_dut1_override/1* *load_leap_seconds_override/1* *offset/3* *save_dut1_entries/1* *save_leap_seconds_entries/1* *supported_range/2* *utc_date_time_to_instant/2* *valid_conversion/3* *valid_instant/1* *valid_scale/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

time_scales_protocol, *date*

object

1.322.2 time_scales_data

Bundled and override data plus constants for UTC/TAI/TT/UT1/TDB/TCG/TCB conversions.

Availability:
`logtalk_load(time_scales(loader))`

Author: Paulo Moura
Version: 0:3:0
Date: 2026-08-08

Compilation flags:
`static, context_switching_calls`

Uses:
logtalk

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *load_leap_seconds_override/1*
  - *clear_leap_seconds_override/0*
  - *leap_seconds_source/1*
  - *leap_seconds_entries/1*
  - *save_leap_seconds_entries/1*
  - *load_dut1_override/1*
  - *clear_dut1_override/0*
  - *dut1_source/1*
  - *dut1_entries/1*
  - *save_dut1_entries/1*
  - *leap_offset_at_utc_unix/2*
  - *leap_effective_date/2*
  - *tt_minus_tai/2*
  - *dut1_offset_at_utc_unix/3*
  - *tdb_minus_tt_approx/3*
  - *tcg_minus_tt_approx/3*
  - *tcb_minus_tdb_approx/3*
  - *tai_minus_utc_for_tai_unix/2*
- *Protected predicates*
- *Private predicates*
  - *leap_override_data_/2*
  - *dut1_override_data_/3*
  - *leap_source_/1*
  - *dut1_source_/1*
- *Operators*

Public predicates

load_leap_seconds_override/1

Loads leap-second override data from a file containing `leap(UnixSeconds,OffsetSeconds).` terms.

Compilation flags:
`static`

Template:
`load_leap_seconds_override(File)`
Mode and number of proofs:
`load_leap_seconds_override(+atom)` - `one`

Remarks:

- Term format: Each term must be of the form `leap(UnixSeconds,OffsetSeconds).`.
- Term ordering: Terms must be sorted by increasing `UnixSeconds`.
- Value constraints: `UnixSeconds` must be an integer greater than or equal to `63072000` and `OffsetSeconds` must be a non-decreasing integer sequence.

---

clear_leap_seconds_override/0

Clears leap-second override data and reverts to bundled data.

Compilation flags:
`static`

Mode and number of proofs:
`clear_leap_seconds_override` - `one`

---

leap_seconds_source/1

Returns the active leap-seconds data source as `bundled` or `override`.

Compilation flags:
`static`

Template:
`leap_seconds_source(Source)`
Mode and number of proofs:
`leap_seconds_source(-atom)` - `one`

---

leap_seconds_entries/1

Returns the active leap-seconds data as an ordered list of `leap(UnixSeconds,OffsetSeconds)` terms.

Compilation flags:
`static`

Template:
`leap_seconds_entries(Entries)`
Mode and number of proofs:
`leap_seconds_entries(-list)` - `one`

---

save_leap_seconds_entries/1

Saves the active leap-seconds data to a file using `leap(UnixSeconds,OffsetSeconds).` terms.

Compilation flags:
`static`

Template:
`save_leap_seconds_entries(File)`
Mode and number of proofs:
`save_leap_seconds_entries(+atom)` - `one`

---

load_dut1_override/1

Loads DUT1 override data from a file containing `dut1(UnixSeconds,Numerator,Denominator).` terms.

Compilation flags:
`static`

Template:
`load_dut1_override(File)`
Mode and number of proofs:
`load_dut1_override(+atom)` - `one`

Remarks:

- Term format: Each term must be of the form `dut1(UnixSeconds,Numerator,Denominator).`.
- Term ordering: Terms must be sorted by increasing `UnixSeconds`.
- Value constraints: `UnixSeconds` and `Numerator` must be integers; `Denominator` must be a positive integer; `UnixSeconds` must be greater than or equal to `63072000`.

---

clear_dut1_override/0

Clears DUT1 override data and reverts to bundled data.

Compilation flags:
`static`

Mode and number of proofs:
`clear_dut1_override` - `one`

---

dut1_source/1

Returns the active DUT1 data source as `bundled` or `override`.

Compilation flags:
`static`

Template:
`dut1_source(Source)`
Mode and number of proofs:
`dut1_source(-atom)` - `one`

---

dut1_entries/1

Returns the active DUT1 data as an ordered list of `dut1(UnixSeconds,Numerator,Denominator)` terms.

Compilation flags:
`static`

Template:
`dut1_entries(Entries)`
Mode and number of proofs:
`dut1_entries(-list)` - `one`

---

save_dut1_entries/1

Saves the active DUT1 data to a file using `dut1(UnixSeconds,Numerator,Denominator).` terms.

Compilation flags:
`static`

Template:
`save_dut1_entries(File)`

Mode and number of proofs:
`save_dut1_entries(+atom)` - `one`

`leap_offset_at_utc_unix/2`

Returns the `TAI-UTC` offset in SI seconds for a given UTC Unix epoch second within the supported range.

Compilation flags:
`static`

Template:
`leap_offset_at_utc_unix(UnixSeconds,OffsetSeconds)`
Mode and number of proofs:
`leap_offset_at_utc_unix(+integer,-integer)` - `zero_or_one`

`leap_effective_date/2`

Enumerates UTC effective dates for `TAI-UTC` offset changes and their resulting offset in SI seconds.

Compilation flags:
`static`

Template:
`leap_effective_date(UTCDateTime,OffsetSeconds)`
Mode and number of proofs:
`leap_effective_date(?compound,?integer)` - `zero_or_more`

`tt_minus_tai/2`

Returns the constant TT minus TAI offset as a rational value `Numerator/Denominator` seconds.

Compilation flags:
`static`

Template:
`tt_minus_tai(Numerator,Denominator)`
Mode and number of proofs:
`tt_minus_tai(-integer,-integer)` - `one`

---

dut1_offset_at_utc_unix/3

Returns DUT1 (UT1−UTC) at a UTC Unix epoch second as a rational value Numerator/Denominator.

Compilation flags:
static

Template:
dut1_offset_at_utc_unix(UnixSeconds,Numerator,Denominator)
Mode and number of proofs:
dut1_offset_at_utc_unix(+integer,-integer,-integer) - zero_or_one

---

tdb_minus_tt_approx/3

Returns an approximate TDB−TT offset in seconds for a TT instant represented by integer seconds and normalized fraction.

Compilation flags:
static

Template:
tdb_minus_tt_approx(TTSeconds,Fraction,OffsetSeconds)
Mode and number of proofs:
tdb_minus_tt_approx(+integer,+compound,-float) - one

---

tcg_minus_tt_approx/3

Returns an approximate TCG−TT offset in seconds for a TT instant represented by integer seconds and normalized fraction.

Compilation flags:
static

Template:
tcg_minus_tt_approx(TTSeconds,Fraction,OffsetSeconds)
Mode and number of proofs:
tcg_minus_tt_approx(+integer,+compound,-float) - one

---

tcb_minus_tdb_approx/3

Returns an approximate `TCB-TDB` offset in seconds for a TDB instant represented by integer seconds and normalized fraction.

Compilation flags:
`static`

Template:
`tcb_minus_tdb_approx(TDBSeconds,Fraction,OffsetSeconds)`
Mode and number of proofs:
`tcb_minus_tdb_approx(+integer,+compound,-float)` - `one`

---

tai_minus_utc_for_tai_unix/2

Returns the `TAI-UTC` offset in SI seconds for a given TAI instant represented as Unix-like integer seconds.

Compilation flags:
`static`

Template:
`tai_minus_utc_for_tai_unix(TAISeconds,OffsetSeconds)`
Mode and number of proofs:
`tai_minus_utc_for_tai_unix(+integer,-integer)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

leap_override_data_/2

Table of leap seconds override data.

Compilation flags:
`dynamic`

Template:
`leap_override_data_(UnixStart,OffsetSeconds)`
Mode and number of proofs:
`leap_override_data_(?integer,?integer)` - `zero_or_more`

---

dut1_override_data_/3

Table of DUT1 override data.

Compilation flags:
`dynamic`

Template:
`dut1_override_data_(UnixStart,Numerator,Denominator)`
Mode and number of proofs:
`dut1_override_data_(?integer,?integer,?integer)` - `zero_or_more`

---

leap_source_/1

Source of leap seconds data (either `override` or `bundled`).

Compilation flags:
`dynamic`

Template:
`leap_source_(Source)`
Mode and number of proofs:
`leap_source_(?atom)` - `zero_or_one`

---

dut1_source_/1

Source of DUT1 data (either `override` or `bundled`).

Compilation flags:
`dynamic`

Template:
`dut1_source_(Source)`
Mode and number of proofs:
`dut1_source_(?atom)` - `zero_or_one`

Operators

(none)
protocol

1.322.3 `time_scales_protocol`

Time scales conversion protocol for UTC, TAI, TT, UT1, TDB, GPS, GST, TCG, and TCB.

Availability:
`logtalk_load(time_scales(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-25

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *valid_scale/1*
  - *supported_range/2*
  - *leap_second_date/2*
  - *load_leap_seconds_override/1*
  - *clear_leap_seconds_override/0*
  - *leap_seconds_source/1*
  - *leap_seconds_entries/1*
  - *save_leap_seconds_entries/1*
  - *load_dut1_override/1*
  - *clear_dut1_override/0*
  - *dut1_source/1*
  - *dut1_entries/1*
  - *save_dut1_entries/1*
  - *utc_date_time_to_instant/2*
  - *valid_instant/1*
  - *check_instant/1*
  - *valid_conversion/3*
  - *check_conversion/3*
  - *instant_to_utc_date_time/2*

    - *convert/4*
    - *check_convert/4*
    - *offset/3*
    - *check_offset/3*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

valid_scale/1

True when the argument is a supported time scale (`utc`, `tai`, `tt`, `ut1`, `tdb`, `gps`, `gst`, `tcg`, or `tcb`).

Compilation flags:
`static`

Template:
`valid_scale(Scale)`
Mode and number of proofs:
`valid_scale(@atom)` - `zero_or_one`

---

supported_range/2

Returns the supported UTC datetime range as `date_time(Year,Month,Day,Hour,Minute,Second)` terms.

Compilation flags:
`static`

Template:
`supported_range(StartUTC,EndUTC)`
Mode and number of proofs:
`supported_range(-compound,-compound)` - `one`

---

leap_second_date/2

Enumerates UTC effective dates for `TAI-UTC` step changes and their resulting offset in SI seconds.

Compilation flags:
`static`

Template:
`leap_second_date(UTCDateTime,OffsetSeconds)`

Mode and number of proofs:

`leap_second_date(?compound,?integer)` - `zero_or_more`

---

load_leap_seconds_override/1

Loads leap-second override data from a user-provided file containing `leap(UnixSeconds,OffsetSeconds).` terms.

Compilation flags:

`static`

Template:

`load_leap_seconds_override(File)`

Mode and number of proofs:

`load_leap_seconds_override(+atom)` - `one`

---

clear_leap_seconds_override/0

Clears any previously loaded leap-second override data and reverts to bundled data.

Compilation flags:

`static`

Mode and number of proofs:

`clear_leap_seconds_override` - `one`

---

leap_seconds_source/1

Returns the active leap-seconds data source as either `bundled` or `override`.

Compilation flags:

`static`

Template:

`leap_seconds_source(Source)`

Mode and number of proofs:

`leap_seconds_source(-atom)` - `one`

---

leap_seconds_entries/1

Returns the active leap-seconds table as an ordered list of `leap(UnixSeconds,OffsetSeconds)` terms.

Compilation flags:
`static`

Template:
`leap_seconds_entries(Entries)`
Mode and number of proofs:
`leap_seconds_entries(-list)` - `one`

---

save_leap_seconds_entries/1

Saves the active leap-seconds table to a file as `leap(UnixSeconds,OffsetSeconds).` terms.

Compilation flags:
`static`

Template:
`save_leap_seconds_entries(File)`
Mode and number of proofs:
`save_leap_seconds_entries(+atom)` - `one`

---

load_dut1_override/1

Loads DUT1 override data from a user-provided file containing `dut1(UnixSeconds,Numerator,Denominator).` terms.

Compilation flags:
`static`

Template:
`load_dut1_override(File)`
Mode and number of proofs:
`load_dut1_override(+atom)` - `one`

---

clear_dut1_override/0

Clears any previously loaded DUT1 override data and reverts to bundled data.

Compilation flags:
`static`

Mode and number of proofs:
`clear_dut1_override` - `one`

---

dut1_source/1

Returns the active DUT1 data source as either `bundled` or `override`.

Compilation flags:
`static`

Template:
`dut1_source(Source)`
Mode and number of proofs:
`dut1_source(-atom)` - `one`

---

dut1_entries/1

Returns the active DUT1 table as an ordered list of `dut1(UnixSeconds,Numerator,Denominator)` terms.

Compilation flags:
`static`

Template:
`dut1_entries(Entries)`
Mode and number of proofs:
`dut1_entries(-list)` - `one`

---

save_dut1_entries/1

Saves the active DUT1 table to a file as `dut1(UnixSeconds,Numerator,Denominator).` terms.

Compilation flags:
`static`

Template:
`save_dut1_entries(File)`

Mode and number of proofs:
`save_dut1_entries(+atom)` - `one`

---

utc_date_time_to_instant/2

Converts a UTC datetime term to an instant represented as `instant(utc,Seconds,fraction(Numerator,Denominator))` where `Seconds` are Unix epoch seconds and the fractional component is normalized.

Compilation flags:
`static`

Template:
`utc_date_time_to_instant(UTCDateTime,Instant)`
Mode and number of proofs:
`utc_date_time_to_instant(+compound,-compound)` - `zero_or_one`

---

valid_instant/1

True when the argument is a valid supported instant term `instant(Scale,Seconds,fraction(Numerator,Denominator))`.

Compilation flags:
`static`

Template:
`valid_instant(Instant)`
Mode and number of proofs:
`valid_instant(+compound)` - `zero_or_one`

---

check_instant/1

Checks an instant term and throws an error instead of failing on invalid input.

Compilation flags:
`static`

Template:
`check_instant(Instant)`
Mode and number of proofs:
`check_instant(+compound)` - `one_or_error`

Exceptions:

`Instant` is a variable:
```
instantiation_error
```
`Instant` is not a valid instant term:
```
domain_error(instant,Instant)
```
`Scale` in `Instant` is neither a variable nor an atom:
```
type_error(atom,Scale)
```
`Scale` in `Instant` is not supported:
```
domain_error(time_scale,Scale)
```
`Seconds` in `Instant` is neither a variable nor an integer:
```
type_error(integer,Seconds)
```
`Seconds` in `Instant` is before supported UTC epoch:
```
domain_error(utc_unix_seconds,Seconds)
```
Fraction term in `Instant` is not of the form `fraction(Numerator,Denominator)`:
```
domain_error(fraction,Fraction)
```
`Numerator` in `Instant` is neither a variable nor an integer:
```
type_error(integer,Numerator)
```
`Denominator` in `Instant` is neither a variable nor an integer:
```
type_error(integer,Denominator)
```
`Denominator` in `Instant` is not positive:
```
domain_error(positive_denominator,Denominator)
```
Normalized fraction in `Instant` is not in the `[0,1[` interval:
```
domain_error(normalized_fraction,Fraction)
```

---

valid_conversion/3

True when an instant is valid for `FromScale` and conversion to `ToScale` is admissible.

Compilation flags:
```
static
```

Template:
```
valid_conversion(Instant,FromScale,ToScale)
```
Mode and number of proofs:
```
valid_conversion(+compound,+atom,+atom) - zero_or_one
```

---

check_conversion/3

Checks a conversion request and throws an error instead of failing on invalid input.

Compilation flags:
```
static
```

Template:
```
check_conversion(Instant,FromScale,ToScale)
```
Mode and number of proofs:

`check_conversion(+compound,+atom,+atom)` - `one_or_error`

Exceptions:

`Instant`, `FromScale`, or `ToScale` is a variable:
`instantiation_error`
`FromScale` is neither a variable nor an atom:
`type_error(atom,FromScale)`
`ToScale` is neither a variable nor an atom:
`type_error(atom,ToScale)`
`FromScale` is not supported:
`domain_error(time_scale,FromScale)`
`ToScale` is not supported:
`domain_error(time_scale,ToScale)`
`Instant` is not a valid instant term:
`domain_error(instant,Instant)`
`Scale` in `Instant` is neither a variable nor an atom:
`type_error(atom,Scale)`
`Scale` in `Instant` is not supported:
`domain_error(time_scale,Scale)`
`Seconds` in `Instant` is neither a variable nor an integer:
`type_error(integer,Seconds)`
`Seconds` in `Instant` is before supported UTC epoch:
`domain_error(utc_unix_seconds,Seconds)`
Fraction term in `Instant` is not of the form `fraction(Numerator,Denominator)`:
`domain_error(fraction,Fraction)`
`Numerator` in `Instant` is neither a variable nor an integer:
`type_error(integer,Numerator)`
`Denominator` in `Instant` is neither a variable nor an integer:
`type_error(integer,Denominator)`
`Denominator` in `Instant` is not positive:
`domain_error(positive_denominator,Denominator)`
Normalized fraction in `Instant` is not in the `[0,1[` interval:
`domain_error(normalized_fraction,Fraction)`
`Instant` scale does not match `FromScale`:
`domain_error(scale_mismatch,instant_scale(InstantScale,FromScale))`

`instant_to_utc_date_time/2`

Converts an UTC instant represented as `instant(utc,Seconds,fraction(Numerator,Denominator))` to a UTC datetime term. Requires zero fractional part.

Compilation flags:

`static`

Template:

`instant_to_utc_date_time(Instant,UTCDateTime)`

Mode and number of proofs:

`instant_to_utc_date_time(+compound,-compound) - zero_or_one`

---

convert/4

Converts an `instant(Scale,Seconds,fraction(Numerator,Denominator))` from `FromScale` to `ToScale`.

Compilation flags:
`static`

Template:
`convert(Instant,FromScale,ToScale,ConvertedInstant)`
Mode and number of proofs:
`convert(+compound,+atom,+atom,-compound) - zero_or_one`

---

check_convert/4

Converts an instant from `FromScale` to `ToScale` and throws an error instead of failing on invalid input.

Compilation flags:
`static`

Template:
`check_convert(Instant,FromScale,ToScale,ConvertedInstant)`
Mode and number of proofs:
`check_convert(+compound,+atom,+atom,-compound) - one_or_error`

Exceptions:
`Instant`, `FromScale`, or `ToScale` is a variable:
`instantiation_error`
`FromScale` is neither a variable nor an atom:
`type_error(atom,FromScale)`
`ToScale` is neither a variable nor an atom:
`type_error(atom,ToScale)`
`FromScale` is not supported:
`domain_error(time_scale,FromScale)`
`ToScale` is not supported:
`domain_error(time_scale,ToScale)`
`Instant` is not a valid instant term:
`domain_error(instant,Instant)`
`Scale` in `Instant` is neither a variable nor an atom:
`type_error(atom,Scale)`
`Scale` in `Instant` is not supported:
`domain_error(time_scale,Scale)`
`Seconds` in `Instant` is neither a variable nor an integer:
`type_error(integer,Seconds)`

`Seconds` in `Instant` is before supported UTC epoch:
`domain_error(utc_unix_seconds,Seconds)`
Fraction term in `Instant` is not of the form `fraction(Numerator,Denominator)`:
`domain_error(fraction,Fraction)`
`Numerator` in `Instant` is neither a variable nor an integer:
`type_error(integer,Numerator)`
`Denominator` in `Instant` is neither a variable nor an integer:
`type_error(integer,Denominator)`
`Denominator` in `Instant` is not positive:
`domain_error(positive_denominator,Denominator)`
Normalized fraction in `Instant` is not in the `[0,1[` interval:
`domain_error(normalized_fraction,Fraction)`
`Instant` scale does not match `FromScale`:
`domain_error(scale_mismatch,instant_scale(InstantScale,FromScale))`

`offset/3`

Returns the offset required to convert an instant to a target scale as `rational(Numerator,Denominator)`.

Compilation flags:
`static`

Template:
`offset(Instant,ToScale,Offset)`
Mode and number of proofs:
`offset(+compound,+atom,-compound)` - `zero_or_one`

`check_offset/3`

Returns the conversion offset and throws an error instead of failing on invalid input.

Compilation flags:
`static`

Template:
`check_offset(Instant,ToScale,Offset)`
Mode and number of proofs:
`check_offset(+compound,+atom,-compound)` - `one_or_error`

Exceptions:
`Instant` or `ToScale` is a variable:
`instantiation_error`
`ToScale` is neither a variable nor an atom:
`type_error(atom,ToScale)`
`ToScale` is not supported:

`domain_error(time_scale,ToScale)`

`Instant` is not a valid instant term:

`domain_error(instant,Instant)`

`Scale` in `Instant` is neither a variable nor an atom:

`type_error(atom,Scale)`

`Scale` in `Instant` is not supported:

`domain_error(time_scale,Scale)`

`Seconds` in `Instant` is neither a variable nor an integer:

`type_error(integer,Seconds)`

`Seconds` in `Instant` is before supported UTC epoch:

`domain_error(utc_unix_seconds,Seconds)`

Fraction term in `Instant` is not of the form `fraction(Numerator,Denominator)`:

`domain_error(fraction,Fraction)`

`Numerator` in `Instant` is neither a variable nor an integer:

`type_error(integer,Numerator)`

`Denominator` in `Instant` is neither a variable nor an integer:

`type_error(integer,Denominator)`

`Denominator` in `Instant` is not positive:

`domain_error(positive_denominator,Denominator)`

Normalized fraction in `Instant` is not in the `[0,1[` interval:

`domain_error(normalized_fraction,Fraction)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

time_scales, *date*

1.323 time_series_protocols

category

1.323.1 forecaster_common

Shared predicates for forecaster diagnostics, dataset validation, differencing, lag construction, error metrics, seasonal adjustment, and baselines.

Availability:
`logtalk_load(time_series_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-18

Compilation flags:
`static`

Implements:
`public` *forecaster_protocol*
Extends:
`public` *options*
Uses:
format
list
numberlist
type

Remarks:
(none)

Inherited public predicates:
check_forecaster/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *forecast/3* *forecaster_options/2* *learn/2* *learn/3* *option/2* *option/3* *print_forecaster/1* *valid_forecaster/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - *forecaster_diagnostics_data/2*
  - *forecaster_export_template/4*
  - *forecaster_term_template/2*
  - *print_forecaster_template/1*
  - *dataset_series/2*
  - *check_series/2*
  - *check_series/3*
  - *check_series_length/3*
  - *check_observation/1*
  - *series_observation_summary/4*
  - *normalize_missing_series/3*
  - *indexed_series_observations/3*
  - *residual_fitted_values/3*

    - *valid_residual_history/5*
    - *base_forecaster_diagnostics/5*
    - *valid_forecaster_metadata/2*
    - *valid_forecaster_metadata/3*
    - *replace_diagnostic/4*
    - *updated_observation_diagnostics/3*
    - *difference_series/2*
    - *integrate_series/3*
    - *lagged_rows/3*
    - *accumulate_forecast_error/4*
    - *forecast_error_metrics/3*
    - *mean_absolute_error/3*
    - *root_mean_squared_error/3*
    - *mean_absolute_percentage_error/3*
    - *naive_forecast/3*
    - *constant_forecast/3*
    - *linear_trend_forecast/4*
    - *check_forecast_horizon/1*
    - *seasonal_naive_forecast/4*
    - *check_frequency/1*
    - *seasonal_autocorrelation_test/4*
    - *classical_seasonal_adjustment/5*
    - *restore_seasonality/5*
- *Private predicates*
    - *sorted_observations/5*
    - *normalize_missing_values/4*
    - *indexed_known_values/5*
    - *check_numeric_residuals/2*
    - *aligned_residual_fits/5*
    - *valid_residual_positions/3*
    - *seasonal_centered_values/5*
    - *seasonal_lag_scores/7*
    - *seasonal_covariance/4*
    - *seasonal_check_method/1*
    - *seasonal_positive_values/1*
    - *seasonal_window_means/5*
    - *seasonal_center_even/2*
    - *seasonal_deviations/4*
    - *seasonal_fold_cycles/3*
    - *seasonal_bucket_means/2*
    - *seasonal_normalize/4*
    - *seasonal_inverse_factors/3*
    - *seasonal_restore_values/5*
    - *seasonal_complete_autocorrelation_test/4*
    - *classical_complete_seasonal_adjustment/5*
    - *seasonal_gap_centered/4*
    - *seasonal_gap_lags/7*

    - *seasonal_gap_covariance/4*
    - *seasonal_gap_windows/5*
    - *seasonal_gap_even/2*
    - *seasonal_gap_deviations/4*
    - *seasonal_accumulate_cycle/4*
    - *seasonal_check_buckets/2*
  - *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

forecaster_diagnostics_data/2

Hook predicate that importing forecaster implementations must define in order to expose diagnostics metadata. A default implementation is provided that assumes the diagnostics list is the last argument of the forecaster term; concrete implementations following that convention do not need to override it.

Compilation flags:

```
static
```

Template:

```
forecaster_diagnostics_data(Forecaster,Diagnostics)
```

Mode and number of proofs:

```
forecaster_diagnostics_data(+compound,-list(compound)) - one
```

---

forecaster_export_template/4

Hook predicate that importing forecaster implementations must define in order to expose the exported forecaster template for a given functor.

Compilation flags:

```
static
```

Template:

```
forecaster_export_template(Dataset,Forecaster,Functor,Template)
```

Mode and number of proofs:

```
forecaster_export_template(+object_identifier,+compound,+atom,-callable) - one
```

---

forecaster_term_template/2

Hook predicate that importing forecaster implementations must define in order to expose the learned forecaster term template used by pretty-printing helpers.

Compilation flags:

```
static
```

Template:

```
forecaster_term_template(Forecaster,Template)
```

Mode and number of proofs:

```
forecaster_term_template(+compound,-callable) - one
```

---

print_forecaster_template/1

Pretty-printing helper predicate used by importing forecaster implementations to show the learned forecaster term template.

Compilation flags:

```
static
```

Template:

```
print_forecaster_template(Forecaster)
```

Mode and number of proofs:

```
print_forecaster_template(+compound) - one
```

---

dataset_series/2

Collects the dataset time-ordered observation values. Checks that the observation indices form a complete, gap-free, 1-based sequence and that the declared series length matches the observed length.

Compilation flags:

```
static
```

Template:

```
dataset_series(Dataset,Series)
```

Mode and number of proofs:

```
dataset_series(+object_identifier,-list) - one_or_error
```

Exceptions:

The dataset contains no observations:

```
domain_error(non_empty_series,Dataset)
```

The observation indices do not form a complete, gap-free, 1-based sequence:

```
domain_error(series_index_sequence,Dataset)
```

An observation index or the declared series length is a variable:

```
instantiation_error
```

The declared series length is neither a variable nor an integer:

```
type_error(integer,DeclaredLength)
```

The declared series length is an integer but is not positive:

```
domain_error(positive_integer,DeclaredLength)
```

The declared and observed series lengths differ:

```
consistency_error(series_length,DeclaredLength,ObservedLength)
```

Index arithmetic raises an evaluation error:

```
evaluation_error(Error)
```

---

check_series/2

Checks that a time series is non-empty and that every observation value is a number.

Compilation flags:

```
static
```

Template:

```
check_series(Dataset,Series)
```

Mode and number of proofs:

```
check_series(+object_identifier,+list) - one_or_error
```

Exceptions:

`Series` is a partial list or a list with an element which is a variable:

```
instantiation_error
```

An element `Value` of the `Series` list is neither a variable nor a number:

```
type_error(number,Value)
```

`Series` is empty:

```
domain_error(non_empty_series,Dataset)
```

---

check_series/3

Checks that a time series is non-empty and that every observation value is of one of the given types.

Compilation flags:

```
static
```

Template:

```
check_series(Dataset,Series,Types)
```

Mode and number of proofs:

```
check_series(+object_identifier,+list,+list) - one_or_error
```

Exceptions:

`Series` is a partial list:

```
instantiation_error
```

An element `Value` of the `Series` list is neither a variable nor a number:

`domain_error(types(Types),Value)`
`Series` is empty:
`domain_error(non_empty_series,Dataset)`

---

check_series_length/3

Checks that a time series has at least the given minimum number of observations.

Compilation flags:
`static`

Template:
`check_series_length(Dataset,Series,MinimumLength)`
Mode and number of proofs:
`check_series_length(+object_identifier,+list,+non_negative_integer)` - `one_or_error`

Exceptions:
`MinimumLength` is a variable:
`instantiation_error`
`MinimumLength` is neither a variable nor an integer:
`type_error(integer,MinimumLength)`
`MinimumLength` is an integer but is negative:
`domain_error(non_negative_integer,MinimumLength)`
`Series` is shorter than `MinimumLength`:
`domain_error(series_length,Dataset)`

---

check_observation/1

Checks that an observation is a number or an unbound variable representing a missing observation.

Compilation flags:
`static`

Template:
`check_observation(Observation)`
Mode and number of proofs:
`check_observation(@term)` - `one_or_error`

Exceptions:
`Observation` is neither a variable nor a number:
`type_error(number,Observation)`

---

series_observation_summary/4

Returns the elapsed length, numeric observation count, and numeric sum of a series. Unbound observations are counted in the length only and are not instantiated. An empty list has zero length, count, and sum.

Compilation flags:

`static`

Template:

`series_observation_summary(Series,Length,ObservedCount,Sum)`

Mode and number of proofs:

`series_observation_summary(+list,-non_negative_integer,-non_negative_integer,-number)` - `one_or_error`

Exceptions:

`Series` is a variable or a partial list:

`instantiation_error`

`Series` is neither a variable nor a list:

`type_error(list,Series)`

An observation is neither a variable nor a number:

`type_error(number,Observation)`

Numeric summation raises an arithmetic evaluation error:

`evaluation_error(Error)`

---

normalize_missing_series/3

Copies numeric-or-unbound observations, sharing a fresh variable among missing positions without binding input variables.

Compilation flags:

`static`

Template:

`normalize_missing_series(Series,Normalized,Missing)`

Mode and number of proofs:

`normalize_missing_series(+list,-list,-term)` - `one_or_error`

Exceptions:

The series is unbound or partial:

`instantiation_error`

The series is not a list:

`type_error(list,Series)`

A known observation is not numeric:

`type_error(number,Value)`

---

indexed_series_observations/3

Collects known numeric values with their original one-based time indices.

Compilation flags:

```
static
```

Template:

```
indexed_series_observations(Series,Indices,Values)
```

Mode and number of proofs:

```
indexed_series_observations(+list,-list(positive_integer),-list(number)) -
one_or_error
```

Exceptions:

The list is unbound or partial:

```
instantiation_error
```

The series is not a list:

```
type_error(list,Series)
```

A known value is not numeric:

```
type_error(number,Value)
```

Index arithmetic fails:

```
evaluation_error(Error)
```

---

residual_fitted_values/3

Reconstructs aligned pre-update fits from numeric-or-unbound observations and residuals after the first known initialization anchor. Missing positions and the anchor have fresh independent unbound output placeholders.

Compilation flags:

```
static
```

Template:

```
residual_fitted_values(Series,Residuals,Values)
```

Mode and number of proofs:

```
residual_fitted_values(+list,+list(number),-list) - one_or_error
```

Exceptions:

A list is unbound or partial, or a residual is unbound:

```
instantiation_error
```

Series or residuals are not lists:

```
type_error(list,List)
```

A known observation or residual is not numeric:

```
type_error(number,Value)
```

The residual count differs from the known count minus the initialization anchor:

```
domain_error(residual_count,Residuals)
```

Fit reconstruction arithmetic fails:

```
evaluation_error(Error)
```

---

valid_residual_history/5

Validates ground parallel numeric residuals and strictly increasing original indices after the initialization anchor. Both lists must match the scored count. The caller validates finiteness; this predicate does not reconstruct error totals or certify historical observations.

Compilation flags:

```
static
```

Template:

```
valid_residual_history(Residuals,Indices,Anchor,Length,ScoredCount)
```

Mode and number of proofs:

```
valid_residual_history(@list,@list,@positive_integer,@positive_integer,
@non_negative_integer) - zero_or_one_or_error
```

Exceptions:

Index or count arithmetic fails:

```
evaluation_error(Error)
```

---

base_forecaster_diagnostics/5

Builds the common part of a forecaster diagnostics metadata list, combined with implementation-specific extra diagnostics terms.

Compilation flags:

```
static
```

Template:

```
base_forecaster_diagnostics(Model,TrainingSeriesLength,Options,ExtraDiagnostics,
Diagnostics)
```

Mode and number of proofs:

```
base_forecaster_diagnostics(+atom,+integer,+list(compound),+list(compound),
-list(compound)) - one
```

---

valid_forecaster_metadata/2

True when diagnostics metadata contains the expected model term.

Compilation flags:

```
static
```

Template:

```
valid_forecaster_metadata(Model,Diagnostics)
```

Mode and number of proofs:

```
valid_forecaster_metadata(+atom,+list(compound)) - zero_or_one
```

---

valid_forecaster_metadata/3

True when diagnostics metadata contains the expected model term and records the given effective options.

Compilation flags:
`static`

Template:
`valid_forecaster_metadata(Model,Options,Diagnostics)`
Mode and number of proofs:
`valid_forecaster_metadata(+atom,+list(compound),+list(compound))` - `zero_or_one`

---

replace_diagnostic/4

Replaces the value of the first unary diagnostic with the given name, preserving order. Fails when no such diagnostic exists.

Compilation flags:
`static`

Template:
`replace_diagnostic(Name,Value,Diagnostics,UpdatedDiagnostics)`
Mode and number of proofs:
`replace_diagnostic(+atom,+term,+list(compound),-list(compound))` - `zero_or_one`

---

updated_observation_diagnostics/3

Updates the training length, update count, observed count, and missing count of validated diagnostics after a numeric or missing observation, preserving metadata order and all other diagnostics. Requires existing non-negative integer counters.

Compilation flags:
`static`

Template:
`updated_observation_diagnostics(Diagnostics,Observation,UpdatedDiagnostics)`
Mode and number of proofs:
`updated_observation_diagnostics(+list(compound),@term,-list(compound))` - `one_or_error`

Exceptions:
`Observation` is neither a variable nor a number:
`type_error(number,Observation)`
Counter arithmetic raises an evaluation error:
`evaluation_error(Error)`

---

difference_series/2

Computes the first-order differences of a series: `Differences[i] = Series[i+1] - Series[i]`. The result has one fewer element than `Series`.

Compilation flags:
`static`

Template:
`difference_series(Series,Differences)`
Mode and number of proofs:
`difference_series(+list(number),-list(number))` - `one_or_error`

Exceptions:
`Series` is empty:
`domain_error(non_empty_series,Series)`

---

integrate_series/3

Reconstructs a series of levels from a series of differences and a starting base value (typically the last observed original value), by cumulative summing. Used to undo `difference_series/2` on forecasted differences.

Compilation flags:
`static`

Template:
`integrate_series(Differences,Base,Series)`
Mode and number of proofs:
`integrate_series(+list(number),+number,-list(number))` - `one`

---

lagged_rows/3

Builds a list of `Lags-Target` rows from a series, where `Lags` is a list of the `Order` most recent values preceding `Target`, most recent first (`[X(t-1), X(t-2), ..., X(t-Order)]-X(t)`). Requires the series to have more than `Order` observations.

Compilation flags:
`static`

Template:
`lagged_rows(Series,Order,Rows)`
Mode and number of proofs:
`lagged_rows(+list(number),+positive_integer,-list(pair))` - `one_or_error`

Exceptions:
`Order` is a variable:

`instantiation_error`
`Order` is neither a variable nor an integer:
`type_error(integer,Order)`
`Order` is an integer but is not positive:
`domain_error(positive_integer,Order)`
`Series` has `Order` or fewer observations:
`domain_error(series_length,Series)`

---

accumulate_forecast_error/4

Adds one numeric actual/prediction error to validated forecast_error_totals(Count, AbsoluteSum, SquaredSum), without retaining observations. Requires a non-negative integer count and non-negative numeric sums; forecast_error_totals(0,0,0) initializes accumulation. Missing observations must be skipped by the caller.

Compilation flags:
`static`

Template:
`accumulate_forecast_error(Actual,Prediction,Totals,UpdatedTotals)`
Mode and number of proofs:
`accumulate_forecast_error(+number,+number,+compound,-compound)` - `one_or_error`

Exceptions:
`Actual` or `Prediction` is a variable:
`instantiation_error`
`Actual` or `Prediction` is not a number:
`type_error(number,Value)`
Error or total arithmetic raises an evaluation error:
`evaluation_error(Error)`

---

forecast_error_metrics/3

Computes MAE and RMSE from validated forecast_error_totals(Count, AbsoluteSum, SquaredSum). Requires non-negative numeric sums and a positive integer count. Does not retain or reconstruct actual/predicted observations.

Compilation flags:
`static`

Template:
`forecast_error_metrics(Totals,MAE,RMSE)`
Mode and number of proofs:
`forecast_error_metrics(+compound,-float,-float)` - `one_or_error`

Exceptions:
The scored count is a variable:

```
instantiation_error
```

The scored count is not an integer:

```
type_error(integer,Count)
```

The scored count is not positive:

```
domain_error(positive_integer,Count)
```

Metric arithmetic raises an evaluation error:

```
evaluation_error(Error)
```

---

mean_absolute_error/3

Computes the mean absolute error (MAE) between actual and predicted value lists of matching length.

Compilation flags:

```
static
```

Template:

```
mean_absolute_error(Actual,Predicted,MAE)
```

Mode and number of proofs:

```
mean_absolute_error(+list(number),+list(number),-float) - one_or_error
```

Exceptions:

An input series is empty:

```
domain_error(non_empty_series,Series)
```

The actual and predicted series have different lengths:

```
consistency_error(same_length,Actual,Predicted)
```

---

root_mean_squared_error/3

Computes the root mean squared error (RMSE) between actual and predicted value lists of matching length.

Compilation flags:

```
static
```

Template:

```
root_mean_squared_error(Actual,Predicted,RMSE)
```

Mode and number of proofs:

```
root_mean_squared_error(+list(number),+list(number),-float) - one_or_error
```

Exceptions:

An input series is empty:

```
domain_error(non_empty_series,Series)
```

The actual and predicted series have different lengths:

```
consistency_error(same_length,Actual,Predicted)
```

---

mean_absolute_percentage_error/3

Computes the mean absolute percentage error (MAPE), as a percentage, between actual and predicted value lists of matching length.

Compilation flags:

`static`

Template:

`mean_absolute_percentage_error(Actual,Predicted,MAPE)`

Mode and number of proofs:

`mean_absolute_percentage_error(+list(number),+list(number),-float)` - `one_or_error`

Exceptions:

An input series is empty:

`domain_error(non_empty_series,Series)`

The actual and predicted series have different lengths:

`consistency_error(same_length,Actual,Predicted)`

An actual value is zero:

`evaluation_error(zero_divisor)`

---

naive_forecast/3

Repeats the last observed series value `Horizon` times (the naive/persistence baseline forecast). A zero horizon returns an empty list.

Compilation flags:

`static`

Template:

`naive_forecast(Series,Horizon,Forecasts)`

Mode and number of proofs:

`naive_forecast(+list(number),+non_negative_integer,-list(number))` - `one_or_error`

Exceptions:

`Horizon` is a variable:

`instantiation_error`

`Horizon` is neither a variable nor an integer:

`type_error(integer,Horizon)`

`Horizon` is an integer but is negative:

`domain_error(non_negative_integer,Horizon)`

---

constant_forecast/3

Repeats a value for the given forecast horizon. A zero horizon returns an empty list.

Compilation flags:
`static`

Template:
`constant_forecast(Value,Horizon,Forecasts)`
Mode and number of proofs:
`constant_forecast(+number,+non_negative_integer,-list(number))` - `one_or_error`

Exceptions:
`Horizon` is a variable:
`instantiation_error`
`Horizon` is neither a variable nor an integer:
`type_error(integer,Horizon)`
`Horizon` is an integer but is negative:
`domain_error(non_negative_integer,Horizon)`

---

linear_trend_forecast/4

Forecasts `Last + Step * Slope` for steps one through the horizon. A zero horizon returns an empty list.

Compilation flags:
`static`

Template:
`linear_trend_forecast(Last,Slope,Horizon,Forecasts)`
Mode and number of proofs:
`linear_trend_forecast(+number,+number,+non_negative_integer,-list(number))` - `one_or_error`

Exceptions:
`Horizon`, `Last`, or `Slope` is a variable:
`instantiation_error`
`Horizon` is neither a variable nor an integer:
`type_error(integer,Horizon)`
`Horizon` is an integer but is negative:
`domain_error(non_negative_integer,Horizon)`
`Last` is neither a variable nor a number:
`type_error(number,Last)`
`Slope` is neither a variable nor a number:
`type_error(number,Slope)`
Trend extrapolation raises an arithmetic evaluation error:
`evaluation_error(Error)`

---

check_forecast_horizon/1

Checks that a forecast horizon is a non-negative integer.

Compilation flags:
`static`

Template:
`check_forecast_horizon(Horizon)`
Mode and number of proofs:
`check_forecast_horizon(+non_negative_integer)` - `one_or_error`

Exceptions:
`Horizon` is a variable:
`instantiation_error`
`Horizon` is neither a variable nor an integer:
`type_error(integer,Horizon)`
`Horizon` is an integer but is negative:
`domain_error(non_negative_integer,Horizon)`

---

seasonal_naive_forecast/4

Repeats the last full seasonal cycle of `Frequency` observations, cycling as needed to cover `Horizon` forecasts (the seasonal naive baseline forecast).

Compilation flags:
`static`

Template:
`seasonal_naive_forecast(Series,Frequency,Horizon,Forecasts)`
Mode and number of proofs:
`seasonal_naive_forecast(+list(number),+positive_integer,+non_negative_integer,-list(number))` - `one_or_error`

Exceptions:
`Frequency` is a variable:
`instantiation_error`
`Frequency` is neither a variable nor an integer:
`type_error(integer,Frequency)`
`Frequency` is an integer but is not positive:
`domain_error(positive_integer,Frequency)`
`Horizon` is a variable:
`instantiation_error`
`Horizon` is neither a variable nor an integer:
`type_error(integer,Horizon)`
`Horizon` is an integer but is negative:
`domain_error(non_negative_integer,Horizon)`

`Series` has fewer than `Frequency` observations:
`domain_error(series_length,Series)`

`check_frequency/1`

Checks that a seasonal frequency is a positive integer.

Compilation flags:
`static`

Template:
`check_frequency(Frequency)`
Mode and number of proofs:
`check_frequency(+positive_integer)` - `one_or_error`

Exceptions:
`Frequency` is a variable:
`instantiation_error`
`Frequency` is neither a variable nor an integer:
`type_error(integer,Frequency)`
`Frequency` is an integer but is not positive:
`domain_error(positive_integer,Frequency)`

`seasonal_autocorrelation_test/4`

Tests original-lag available-pair autocorrelations using the known observation count and requiring two known pairs at every tested lag.

Compilation flags:
`static`

Template:
`seasonal_autocorrelation_test(Series,Frequency,Statistic,Seasonal)`
Mode and number of proofs:
`seasonal_autocorrelation_test(+list,+positive_integer,-number,-boolean)` - `one_or_error`

Exceptions:
The series or frequency is unbound:
`instantiation_error`
A series value is not numeric:
`type_error(number,Value)`
The series is not a list:
`type_error(list,Series)`
The series is empty:

```
domain_error(non_empty_series,Series)
```

The frequency is not an integer:

```
type_error(integer,Frequency)
```

The frequency is not positive:

```
domain_error(positive_integer,Frequency)
```

Seasonality test arithmetic fails:

```
evaluation_error(Error)
```

---

classical_seasonal_adjustment/5

Deseasonalizes known values using complete centered windows and normalized phase factors, preserving missing observations as unbound variables.

Compilation flags:

```
static
```

Template:

```
classical_seasonal_adjustment(Series,Frequency,Method,AdjustedSeries,Factors)
```

Mode and number of proofs:

```
classical_seasonal_adjustment(+list,+positive_integer,+atom,-list,-list(number))
- one_or_error
```

Exceptions:

The series, frequency, or method is unbound:

```
instantiation_error
```

A value is not numeric:

```
type_error(number,Value)
```

The series is not a list:

```
type_error(list,Series)
```

The series is empty:

```
domain_error(non_empty_series,Series)
```

The frequency is not an integer:

```
type_error(integer,Frequency)
```

The frequency is not positive:

```
domain_error(positive_integer,Frequency)
```

The frequency is less than two:

```
domain_error(seasonal_frequency,Frequency)
```

The series has fewer than two cycles:

```
domain_error(series_length,Series)
```

The method is invalid:

```
domain_error(seasonal_adjustment_method,Method)
```

A phase has no valid centered-window deviation:

```
domain_error(insufficient_seasonal_phase_observations,Phase)
```

Multiplicative data or factors are not positive:

```
domain_error(positive_number,Value)
```

Seasonal adjustment arithmetic fails:

```
evaluation_error(Error)
```

restore_seasonality/5

Adds or multiplies a seasonal cycle, advancing phase through missing positions without instantiating them.

Compilation flags:

```
static
```

Template:

```
restore_seasonality(Method,Factors,StartPhase,Values,RestoredValues)
```

Mode and number of proofs:

```
restore_seasonality(+atom,+list(number),+positive_integer,+list,-list) -
one_or_error
```

Exceptions:

The method, phase, factor, or values list is unbound:

```
instantiation_error
```

A factor or value is not numeric:

```
type_error(number,Value)
```

Factors or values are not lists:

```
type_error(list,List)
```

The factors are empty:

```
domain_error(non_empty_series,Factors)
```

The method is invalid:

```
domain_error(seasonal_adjustment_method,Method)
```

The phase is not an integer:

```
type_error(integer,StartPhase)
```

The phase is not positive:

```
domain_error(positive_integer,StartPhase)
```

The phase exceeds the cycle length:

```
domain_error(seasonal_phase,StartPhase)
```

A multiplicative factor is not positive:

```
domain_error(positive_number,Value)
```

Restoration arithmetic fails:

```
evaluation_error(Error)
```

Private predicates

sorted_observations/5

Validates sorted indices without binding them while collecting values and counting observations.

Compilation flags:

```
static
```

Template:

```
sorted_observations(Pairs,Dataset,ExpectedIndex,Length,Series)
```

Mode and number of proofs:

```
sorted_observations(@list(pair),+object_identifier,+positive_integer,
-non_negative_integer,-list) - one_or_error
```

Exceptions:

An observation index is a variable:

```
instantiation_error
```

The observation indices are not a gap-free 1-based integer sequence:

```
domain_error(series_index_sequence,Dataset)
```

Index arithmetic raises an evaluation error:

```
evaluation_error(Error)
```

---

normalize_missing_values/4

Copies known numbers and replaces unbound inputs with a fresh shared variable.

Compilation flags:

```
static
```

Template:

```
normalize_missing_values(Series,Missing,Context,Normalized)
```

Mode and number of proofs:

```
normalize_missing_values(+list,-term,+term,-list) - one_or_error
```

Exceptions:

A known observation is not numeric:

```
type_error(number,Value)
```

---

indexed_known_values/5

Extracts known observations while retaining elapsed positions.

Compilation flags:

```
static
```

Template:

```
indexed_known_values(Series,Index,Context,Indices,Values)
```

Mode and number of proofs:

```
indexed_known_values(+list,+positive_integer,+term,-list,-list) - one_or_error
```

Exceptions:

A known value is not numeric:

```
type_error(number,Value)
```

Index arithmetic fails:

```
evaluation_error(Error)
```

---

check_numeric_residuals/2

Checks residual numbers without completing lists or binding entries.

Compilation flags:

```
static
```

Template:

```
check_numeric_residuals(Residuals,Context)
```

Mode and number of proofs:

```
check_numeric_residuals(+list,+term) - one_or_error
```

Exceptions:

A residual is unbound:

```
instantiation_error
```

A residual is not numeric:

```
type_error(number,Value)
```

---

aligned_residual_fits/5

Consumes residuals only at known positions after the initialization anchor, failing on a count mismatch.

Compilation flags:

```
static
```

Template:

```
aligned_residual_fits(Series,Residuals,AnchorSeen,Context,Values)
```

Mode and number of proofs:

```
aligned_residual_fits(+list,+list(number),+boolean,+term,-list) -
zero_or_one_or_error
```

Exceptions:

A known observation is not numeric:

```
type_error(number,Value)
```

Fit reconstruction arithmetic fails:

```
evaluation_error(Error)
```

---

valid_residual_positions/3

Checks original residual positions in strict chronological order within the elapsed length.

Compilation flags:

`static`

Template:

`valid_residual_positions(Indices,Previous,Length)`

Mode and number of proofs:

`valid_residual_positions(+list(integer),+integer,+positive_integer)` - `zero_or_one_or_error`

Exceptions:

Index comparison fails:

`evaluation_error(Error)`

---

seasonal_centered_values/5

Centers validated observations and accumulates squared deviations.

Compilation flags:

`static`

Template:

`seasonal_centered_values(Series,Mean,Centered,Variance0,Variance)`

Mode and number of proofs:

`seasonal_centered_values(+list(number),+number,-list(number),+number,-number)` - `one_or_error`

Exceptions:

Centering arithmetic fails:

`evaluation_error(Error)`

---

seasonal_lag_scores/7

Accumulates earlier squared autocorrelations and the seasonal-lag score.

Compilation flags:

`static`

Template:

`seasonal_lag_scores(Lag,Frequency,Centered,Variance,Squares0,Squares,Last)`

Mode and number of proofs:

`seasonal_lag_scores(+positive_integer,+positive_integer,+list(number),+number,+number,-number,-number)` - `one_or_error`

Exceptions:

Autocorrelation arithmetic fails:

```
evaluation_error(Error)
```

---

seasonal_covariance/4

Accumulates products of aligned centered observations.

Compilation flags:

```
static
```

Template:

```
seasonal_covariance(Suffix,Series,Covariance0,Covariance)
```

Mode and number of proofs:

```
seasonal_covariance(+list(number),+list(number),+number,-number) - one_or_error
```

Exceptions:

Covariance arithmetic fails:

```
evaluation_error(Error)
```

---

seasonal_check_method/1

Checks the seasonal adjustment method without binding it.

Compilation flags:

```
static
```

Template:

```
seasonal_check_method(Method)
```

Mode and number of proofs:

```
seasonal_check_method(@term) - one_or_error
```

Exceptions:

The method is unbound:

```
instantiation_error
```

The method is invalid:

```
domain_error(seasonal_adjustment_method,Method)
```

---

seasonal_positive_values/1

Requires strictly positive validated numeric values.

Compilation flags:
`static`

Template:
`seasonal_positive_values(Values)`
Mode and number of proofs:
`seasonal_positive_values(+list(number))` - `one_or_error`

Exceptions:
A value is not positive:
`domain_error(positive_number,Value)`
Numeric comparison fails:
`evaluation_error(Error)`

---

seasonal_window_means/5

Computes successive moving averages with a rolling sum.

Compilation flags:
`static`

Template:
`seasonal_window_means(Entering,Leaving,Frequency,Sum,Means)`
Mode and number of proofs:
`seasonal_window_means(+list(number),+list(number),+positive_integer,+number,-list(number))` - `one_or_error`

Exceptions:
Moving-average arithmetic fails:
`evaluation_error(Error)`

---

seasonal_center_even/2

Centers even-period moving averages using adjacent means.

Compilation flags:
`static`

Template:
`seasonal_center_even(Means,Centered)`
Mode and number of proofs:

```
seasonal_center_even(+list(number),-list(number)) - one_or_error
```

Exceptions:

Centering arithmetic fails:

```
evaluation_error(Error)
```

---

seasonal_deviations/4

Computes interior differences or ratios against centered trend estimates.

Compilation flags:

```
static
```

Template:

```
seasonal_deviations(Trends,Values,Method,Deviations)
```

Mode and number of proofs:

```
seasonal_deviations(+list(number),+list(number),+atom,-list(number)) -
one_or_error
```

Exceptions:

A multiplicative trend is not positive:

```
domain_error(positive_number,Value)
```

Detrending arithmetic fails:

```
evaluation_error(Error)
```

---

seasonal_fold_cycles/3

Accumulates complete and partial cycles into phase buckets.

Compilation flags:

```
static
```

Template:

```
seasonal_fold_cycles(Values,Buckets0,Buckets)
```

Mode and number of proofs:

```
seasonal_fold_cycles(+list,+list(compound),-list(compound)) - one_or_error
```

Exceptions:

Phase accumulation fails:

```
evaluation_error(Error)
```

---

seasonal_bucket_means/2

Computes phase means from nonempty phase totals.

Compilation flags:
`static`

Template:
`seasonal_bucket_means(Buckets,Means)`
Mode and number of proofs:
`seasonal_bucket_means(+list(compound),-list(number))` - `one_or_error`

Exceptions:
Phase-mean arithmetic fails:
`evaluation_error(Error)`

---

seasonal_normalize/4

Normalizes phase factors to zero or unit mean.

Compilation flags:
`static`

Template:
`seasonal_normalize(Values,Method,Mean,Factors)`
Mode and number of proofs:
`seasonal_normalize(+list(number),+atom,+number,-list(number))` - `one_or_error`

Exceptions:
A multiplicative factor is not positive:
`domain_error(positive_number,Value)`
Normalization arithmetic fails:
`evaluation_error(Error)`

---

seasonal_inverse_factors/3

Negates additive factors or reciprocates positive multiplicative factors.

Compilation flags:
`static`

Template:
`seasonal_inverse_factors(Factors,Method,Inverses)`
Mode and number of proofs:
`seasonal_inverse_factors(+list(number),+atom,-list(number))` - `one_or_error`

Exceptions:

Factor inversion fails:

`evaluation_error(Error)`

---

seasonal_restore_values/5

Restores validated values while cycling through phase factors.

Compilation flags:

`static`

Template:

`seasonal_restore_values(Values,Method,Factors,Cycle,Restored)`

Mode and number of proofs:

`seasonal_restore_values(+list,+atom,+list(number),+list(number),-list)` - `one_or_error`

Exceptions:

Restoration arithmetic fails:

`evaluation_error(Error)`

---

seasonal_complete_autocorrelation_test/4

Computes the original complete-data seasonality statistic for validated inputs.

Compilation flags:

`static`

Template:

`seasonal_complete_autocorrelation_test(Series,Frequency,Statistic,Seasonal)`

Mode and number of proofs:

`seasonal_complete_autocorrelation_test(+list(number),+positive_integer,-number,-boolean)` - `one_or_error`

Exceptions:

Autocorrelation arithmetic fails:

`evaluation_error(Error)`

---

classical_complete_seasonal_adjustment/5

Computes classical seasonal factors for validated complete data.

Compilation flags:

```
static
```

Template:

```
classical_complete_seasonal_adjustment(Series,Frequency,Method,Adjusted,Factors)
```

Mode and number of proofs:

```
classical_complete_seasonal_adjustment(+list(number),+positive_integer,+atom,
-list(number),-list(number)) - one_or_error
```

Exceptions:

Multiplicative data, trends, or factors are not positive:

```
domain_error(positive_number,Value)
```

Seasonal arithmetic fails:

```
evaluation_error(Error)
```

---

seasonal_gap_centered/4

Centers known values without removing gaps.

Compilation flags:

```
static
```

Template:

```
seasonal_gap_centered(Series,Mean,Centered,Variance)
```

Mode and number of proofs:

```
seasonal_gap_centered(+list,+number,-list,-number) - one_or_error
```

Exceptions:

Centering arithmetic fails:

```
evaluation_error(Error)
```

---

seasonal_gap_lags/7

Accumulates original-lag scores, failing when fewer than two pairs are available.

Compilation flags:

```
static
```

Template:

```
seasonal_gap_lags(Lag,Frequency,Centered,Variance,Squares0,Squares,Last)
```

Mode and number of proofs:

```
seasonal_gap_lags(+positive_integer,+positive_integer,+list,+number,+number,
-number,-number) - zero_or_one_or_error
```

Exceptions:

Autocorrelation arithmetic fails:

```
evaluation_error(Error)
```

---

seasonal_gap_covariance/4

Sums centered products and counts pairs with both positions known.

Compilation flags:

```
static
```

Template:

```
seasonal_gap_covariance(Suffix,Series,Covariance,Pairs)
```

Mode and number of proofs:

```
seasonal_gap_covariance(+list,+list,-number,-non_negative_integer) - one_or_error
```

Exceptions:

Covariance arithmetic fails:

```
evaluation_error(Error)
```

---

seasonal_gap_windows/5

Computes moving averages with rolling sums and missing counts, marking incomplete windows.

Compilation flags:

```
static
```

Template:

```
seasonal_gap_windows(Entering,Leaving,Frequency,Window,Means)
```

Mode and number of proofs:

```
seasonal_gap_windows(+list,+list,+positive_integer,+compound,-list) - one_or_error
```

Exceptions:

Moving-average arithmetic fails:

```
evaluation_error(Error)
```

---

seasonal_gap_even/2

Centers adjacent complete windows, requiring the full even-period support.

Compilation flags:
```
static
```

Template:
```
seasonal_gap_even(Means,Trends)
```
Mode and number of proofs:
```
seasonal_gap_even(+list,-list) - one_or_error
```

Exceptions:
Centering arithmetic fails:
```
evaluation_error(Error)
```

---

seasonal_gap_deviations/4

Computes only known deviations with complete trend support, retaining phase positions.

Compilation flags:
```
static
```

Template:
```
seasonal_gap_deviations(Trends,Values,Method,Deviations)
```
Mode and number of proofs:
```
seasonal_gap_deviations(+list,+list,+atom,-list) - one_or_error
```

Exceptions:
A multiplicative trend is not positive:
```
domain_error(positive_number,Value)
```
Detrending arithmetic fails:
```
evaluation_error(Error)
```

---

seasonal_accumulate_cycle/4

Advances a phase bucket for every position, skipping missing contributions.

Compilation flags:
```
static
```

Template:
```
seasonal_accumulate_cycle(Values,Buckets,Updated,Rest)
```
Mode and number of proofs:

```
seasonal_accumulate_cycle(+list,+list(compound),-list(compound),-list) -
one_or_error
```

Exceptions:

Phase accumulation fails:

```
evaluation_error(Error)
```

---

seasonal_check_buckets/2

Reports the lowest absolute phase lacking an estimable seasonal deviation.

Compilation flags:

```
static
```

Template:

```
seasonal_check_buckets(Buckets,Phase)
```

Mode and number of proofs:

```
seasonal_check_buckets(+list(compound),+positive_integer) - one_or_error
```

Exceptions:

A phase has no valid deviation:

```
domain_error(insufficient_seasonal_phase_observations,Phase)
```

Phase arithmetic fails:

```
evaluation_error(Error)
```

Operators

(none)

protocol

1.323.2 forecaster_protocol

Protocol for time series forecasting models.

Availability:

```
logtalk_load(time_series_protocols(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-14

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *learn/3*
  - *learn/2*
  - *forecast/3*
  - *check_forecaster/1*
  - *valid_forecaster/1*
  - *diagnostics/2*
  - *diagnostic/2*
  - *forecaster_options/2*
  - *export_to_clauses/4*
  - *export_to_file/4*
  - *print_forecaster/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/3

Learns a forecasting model from the given time series dataset object using the specified options.

Compilation flags:
`static`

Template:
`learn(Dataset,Forecaster,Options)`
Mode and number of proofs:
`learn(+object_identifier,-compound,+list(compound))` - `one_or_error`

Exceptions:
`Dataset` is a variable:
`instantiation_error`
`Dataset` is neither a variable nor a valid object identifier:
`type_error(object_identifier,Dataset)`
`Dataset` is a valid object identifier but not an existing object:
`existence_error(object,Dataset)`
`Options` is a variable:
`instantiation_error`

`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

---

learn/2

Learns a forecasting model from the given time series dataset object using default options.

Compilation flags:
`static`

Template:
`learn(Dataset,Forecaster)`
Mode and number of proofs:
`learn(+object_identifier,-compound)` - `one_or_error`

Exceptions:
`Dataset` is a variable:
`instantiation_error`
`Dataset` is neither a variable nor a valid object identifier:
`type_error(object_identifier,Dataset)`
`Dataset` is a valid object identifier but not an existing object:
`existence_error(object,Dataset)`

---

forecast/3

Forecasts the next `Horizon` values beyond the training series using the learned forecasting model. A zero horizon returns an empty list.

Compilation flags:
`static`

Template:
`forecast(Forecaster,Horizon,Forecasts)`
Mode and number of proofs:
`forecast(+compound,+non_negative_integer,-list(number))` - `one_or_error`

Exceptions:
`Forecaster` is a variable:
`instantiation_error`

`Forecaster` is not a valid forecaster:
`domain_error(forecaster,Forecaster)`
`Horizon` is a variable:
`instantiation_error`
`Horizon` is neither a variable nor an integer:
`type_error(integer,Horizon)`
`Horizon` is an integer but is negative:
`domain_error(non_negative_integer,Horizon)`

---

check_forecaster/1

Checks that a learned forecaster term is structurally valid for the receiving implementation. Throws an exception when the term is not a valid forecaster representation.

Compilation flags:
`static`

Template:
`check_forecaster(Forecaster)`
Mode and number of proofs:
`check_forecaster(@compound)` - `one_or_error`

Exceptions:
`Forecaster` is a variable:
`instantiation_error`
`Forecaster` is neither a variable nor a valid forecaster:
`domain_error(forecaster,Forecaster)`

---

valid_forecaster/1

True when a learned forecaster term is structurally valid for the receiving implementation. Succeeds iff `check_forecaster/1` succeeds without throwing an exception.

Compilation flags:
`static`

Template:
`valid_forecaster(Forecaster)`
Mode and number of proofs:
`valid_forecaster(@compound)` - `zero_or_one`

---

diagnostics/2

Returns diagnostics metadata for a learned forecaster.

Compilation flags:
```
static
```

Template:
```
diagnostics(Forecaster,Diagnostics)
```
Mode and number of proofs:
```
diagnostics(+compound,-list(compound)) - one
```

---

diagnostic/2

Enumerates individual diagnostics metadata terms for a learned forecaster.

Compilation flags:
```
static
```

Template:
```
diagnostic(Forecaster,Diagnostic)
```
Mode and number of proofs:
```
diagnostic(+compound,?compound) - zero_or_more
```

---

forecaster_options/2

Returns the effective training options recorded in a learned forecaster diagnostics metadata.

Compilation flags:
```
static
```

Template:
```
forecaster_options(Forecaster,Options)
```
Mode and number of proofs:
```
forecaster_options(+compound,-list(compound)) - one
```

---

export_to_clauses/4

Converts a forecaster into a list of predicate clauses. `Functor` is the functor for the generated predicate clauses. When exporting a serialized forecaster term, a noun such as `forecaster` or `model` is usually clearer than a verb such as `forecast`.

Compilation flags:

`static`

Template:

`export_to_clauses(Dataset,Forecaster,Functor,Clauses)`

Mode and number of proofs:

`export_to_clauses(+object_identifier,+compound,+atom,-list(clause))` - `one_or_error`

Exceptions:

`Forecaster` or `Functor` is a variable:

`instantiation_error`

`Forecaster` is not a valid baseline forecaster:

`domain_error(forecaster,Forecaster)`

`Functor` is neither a variable nor an atom:

`type_error(atom,Functor)`

---

export_to_file/4

Exports a forecaster to a file. `Functor` is the functor for the generated predicate clauses. When exporting a serialized forecaster term, a noun such as `forecaster` or `model` is usually clearer than a verb such as `forecast`.

Compilation flags:

`static`

Template:

`export_to_file(Dataset,Forecaster,Functor,File)`

Mode and number of proofs:

`export_to_file(+object_identifier,+compound,+atom,+atom)` - `one_or_error`

Exceptions:

`Forecaster`, `Functor`, or `File`:

`instantiation_error`

`Forecaster` is not a valid forecaster:

`domain_error(forecaster,Forecaster)`

`Functor` is neither a variable nor an atom:

`type_error(atom,Functor)`

`File` is not a valid file specification:

`type_error(source_sink,File)`

The output location does not exist:

`existence_error(source_sink,File)`

The output file cannot be opened:

`permission_error(open,source_sink,File)`

The output stream cannot be written to:
`permission_error(output,stream,Stream)`

`print_forecaster/1`

Prints a forecaster to the current output stream in a human-readable format.

Compilation flags:
`static`

Template:
`print_forecaster(Forecaster)`

Mode and number of proofs:
`print_forecaster(+compound)` - `one_or_error`

Exceptions:
`Forecaster` is a variable:
`instantiation_error`
`Forecaster` is not a valid baseline forecaster:
`domain_error(forecaster,Forecaster)`
The current stream cannot be written to:
`permission_error(output,stream,Stream)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

time_series_dataset_protocol

protocol

1.323.3 time_series_dataset_protocol

Protocol for datasets used with time series forecasting algorithms.

Availability:
`logtalk_load(time_series_protocols(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-27

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *observation/2*
  - *series_length/1*
  - *frequency/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

observation/2

Enumerates by backtracking the time-ordered observations. `Index` is the 1-based position of the observation in the series and `Value` is its numeric value.

Compilation flags:
`static`

Template:
`observation(Index,Value)`
Mode and number of proofs:
`observation(-integer,-number)` - `zero_or_more`

series_length/1

Returns the number of observations in the series. The declared length must match the number of observations enumerated by `observation/2`.

Compilation flags:
`static`

Template:
`series_length(Length)`
Mode and number of proofs:
`series_length(-positive_integer)` - `one`

---

frequency/1

Returns the positive number of observations per seasonal cycle (e.g. 12 for monthly data with yearly seasonality). Fails when the series has no declared seasonal period.

Compilation flags:
`static`

Template:
`frequency(Frequency)`
Mode and number of proofs:
`frequency(-positive_integer)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.324 time_series_regression

object

1.324.1 time_series_regression

Autoregressive (AR) time series forecaster fitted by least squares, with optional intercept, optional differencing (ARI models), automatic order selection using information criteria, and support for missing observations represented as unbound variables.

Availability:

```
logtalk_load(time_series_regression(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-10-14

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public forecaster_common
```

Uses:

format
integer
linear_algebra
list
type
univariate_distributions(Random)

Remarks:

(none)

Inherited public predicates:

check_forecaster/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *forecast/3* *forecaster_options/2* *learn/2* *learn/3* *option/2* *option/3* *print_forecaster/1* *valid_forecaster/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *update/3*
  - *forecast_interval/5*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

update/3

Returns a new forecaster after appending one observation. Observation may be left an unbound variable to represent a missing observation; see update/4.

Compilation flags:

static

Template:

update(Forecaster,Observation,UpdatedForecaster)

Mode and number of proofs:

update(+compound,@number,-compound) - one_or_error

Exceptions:

Forecaster is a variable:

instantiation_error

Forecaster is neither a variable nor a valid forecaster:

domain_error(forecaster,Forecaster)

Observation is neither a variable nor a number:

type_error(number,Observation)

---

forecast_interval/5

Computes analytic prediction interval bounds for the next Horizon forecasts of a learned forecaster, assuming independent, identically distributed, zero-mean Gaussian innovations and treating the fitted coefficients as known. The h-step forecast error variance is the residual variance times the sum of the squared psi weights psi(0)..psi(h-1) of the (integrated) model, and the bounds are the point forecast plus or minus the standard normal quantile for the requested confidence times the forecast error standard deviation. The residual variance is the sum of squared errors divided by the residual degrees of freedom (the number of scored observations minus the design matrix rank). A zero horizon returns two empty lists.

Compilation flags:

static

Template:

forecast_interval(Forecaster,Horizon,Lower,Upper,Options)

Mode and number of proofs:

forecast_interval(+compound,+non_negative_integer,-list(number),-list(number),
+list(compound)) - one_or_error

Exceptions:

Forecaster is a variable:

instantiation_error

Forecaster is neither a variable nor a valid forecaster:

domain_error(forecaster,Forecaster)

Horizon is a variable:

instantiation_error

Horizon is neither a variable nor an integer:

`type_error(integer,Horizon)`
`Horizon` is an integer but is negative:
`domain_error(non_negative_integer,Horizon)`
`Options` is a variable or a partial list:
`instantiation_error`
`Options` is neither a variable nor a list:
`type_error(list,Options)`
An option is a variable or has an unbound argument:
`instantiation_error`
An option is neither a variable nor a compound term:
`type_error(compound,Option)`
An option is a compound term but is not a valid interval option:
`domain_error(option,Option)`
`Horizon` is positive and the forecaster has no residual degrees of freedom (an exactly determined fit):
`domain_error(residual_degrees_of_freedom,Forecaster)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

forecaster_protocol, *time_series_dataset_protocol*

1.325 timeout

object

1.325.1 `timeout`

Predicates for calling goal with a time limit.

Availability:
`logtalk_load(timeout(loader))`

Author: Paulo Moura
Version: 0:11:0
Date: 2026-10-28

Compilation flags:

```
static, context_switching_calls
```

Dependencies:
(none)

Remarks:

- Supported backend Prolog systems: B-Prolog, ECLiPSe, SICStus Prolog, Shumway Prolog, SWI-Prolog, Trealla Prolog, XSB, XVM, and YAP.

Inherited public predicates:
(none)

- *Public predicates*
  - *call_with_timeout/2*
  - *call_with_timeout/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

call_with_timeout/2

Calls a goal deterministically with the given time limit (expressed in seconds). Note that the goal may fail or throw an error before exhausting the time limit.

Compilation flags:
```
static
```

Template:
```
call_with_timeout(Goal,Timeout)
```
Meta-predicate template:
```
call_with_timeout(0,*)
```
Mode and number of proofs:
```
call_with_timeout(+callable,+positive_number) - zero_or_one
```

Exceptions:
Goal does not complete in the allowed time:
```
timeout(Goal)
```

`call_with_timeout/3`

Calls a goal deterministically with the given time limit (expressed in seconds) returning a reified result: `true`, `fail`, `timeout`, or `error(Error)`.

Compilation flags:
`static`

Template:
`call_with_timeout(Goal,Timeout,Result)`
Meta-predicate template:
`call_with_timeout(0,*,*)`
Mode and number of proofs:
`call_with_timeout(+callable,+positive_number,--atom)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.326 tle_orbits

object

1.326.1 `tle_orbits`

Parser for Two-Line Element sets with an approximate portable propagator featuring automatic branching between near-earth and deep-space variants, a legacy two-body fallback model, and ground-track sampling.

Availability:
`logtalk_load(tle_orbits(loader))`

Author: Paulo Moura
Version: 1:2:1
Date: 2026-08-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *tle_orbits_protocol*

Uses:

crs_projections
date
iso8601
list
reader
time

Remarks:

(none)

Inherited public predicates:

ground_track/5 *ground_track/6* *parse/2* *parse_lines/4* *propagate/3* *propagate/4* *propagate/5* *propagate_state/4* *propagate_state/5*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.326.2 `tle_orbits_protocol`

Protocol for parsing Two-Line Element sets and performing approximate portable orbit propagation with near-earth and deep-space variants plus ground-track sampling.

Availability:

`logtalk_load(tle_orbits(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *parse_lines/4*
  - *propagate/3*
  - *propagate/4*
  - *propagate/5*
  - *propagate_state/4*
  - *propagate_state/5*
  - *ground_track/5*
  - *ground_track/6*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses one or more TLE records from a source specification into canonical `tle(...)` terms. Supported source specifications are `atom(Atom)`, `chars(List)`, `codes(List)`, `stream(Stream)`, and `file(Path)`.

Compilation flags:
`static`

Template:
`parse(Source,TLEs)`

Mode and number of proofs:
`parse(++compound,--list(compound))` - `one_or_error`

Exceptions:
`Source` is a variable or contains a variable source argument:
`instantiation_error`
`Source` is neither a variable nor a valid source specification:
`domain_error(tle_source,Source)`
The parsed non-blank lines cannot be grouped into valid TLE records:
`domain_error(tle_records,Lines)`

parse_lines/4

Parses a single TLE record given an optional name atom or the atom `none` plus the two raw TLE lines.

Compilation flags:
`static`

Template:
`parse_lines(Name,Line1,Line2,TLE)`
Mode and number of proofs:
`parse_lines(++nonvar,++atom,++atom,--compound)` - `one_or_error`

Exceptions:
`Name` is a variable:
`instantiation_error`
`Name` is neither the atom `none` nor an atom name:
`domain_error(tle_name,Name)`
`Line1` is a variable:
`instantiation_error`
`Line1` is neither a variable nor an atom:
`domain_error(tle_line,Line1)`
`Line2` is a variable:
`instantiation_error`
`Line2` is neither a variable nor an atom:
`domain_error(tle_line,Line2)`
The given name and lines do not form a valid TLE record because of invalid syntax, checksum, field encoding, or orbital element ranges:
`domain_error(tle_lines,[Name,Line1,Line2])`

propagate/3

Propagates a parsed TLE using the default `approximate` model and returns a `geographic(Latitude,Longitude,Height)` coordinate. Supported time specifications are `date_time(Year,Month,Day,Hours,Minutes,Seconds)`, `julian_date(JulianDate)`, and `offset_seconds(SecondsSinceEpoch)`.

Compilation flags:
`static`

Template:
`propagate(TLE,Time,Coordinate)`
Mode and number of proofs:
`propagate(++compound,++nonvar,--compound)` - `one_or_error`

Exceptions:
`Time` is not one of the supported time specifications with the required numeric components:

domain_error(tle_time,Time)

---

propagate/4

Propagates a parsed TLE using the default approximate model and returns coordinates in the requested frame. Supported frames are eci returning eci(X,Y,Z), ecef returning ecef(X,Y,Z), and wgs84_3d returning geographic(Latitude, Longitude,Height).

Compilation flags:

static

Template:

propagate(TLE,Time,Frame,Coordinate)

Mode and number of proofs:

propagate(++compound,++nonvar,++atom,--compound) - one_or_error

Exceptions:

Frame is not one of the supported reference frame identifiers:

domain_error(tle_reference_frame,Frame)

Time is not one of the supported time specifications with the required numeric components:

domain_error(tle_time,Time)

---

propagate/5

Propagates a parsed TLE using the requested propagation model and returns coordinates in the requested frame. Supported models are approximate for automatic near-earth versus deep-space dispatch, approximate_near_earth for the low-period approximate branch with J2 secular and short-period corrections plus low-order B* drag handling, approximate_deep_space for the dedicated deep-space approximate branch with resonance-aware long-period corrections, and two_body for the legacy Keplerian approximation.

Compilation flags:

static

Template:

propagate(TLE,Time,Frame,Model,Coordinate)

Mode and number of proofs:

propagate(++compound,++nonvar,++atom,++atom,--compound) - one_or_error

Exceptions:

Frame is not one of the supported reference frame identifiers:

domain_error(tle_reference_frame,Frame)

Model is not one of the supported propagation model identifiers:

domain_error(tle_propagation_model,Model)

Time is not one of the supported time specifications with the required numeric components:

domain_error(tle_time,Time)

---

propagate_state/4

Propagates a parsed TLE using the default `approximate` model and returns `state(Position,Velocity)` in the requested frame. Velocity is derived directly from the propagated orbital elements in ECI and analytically transformed to the requested frame. Supported frames are `eci` returning `state(eci(X,Y,Z), eci(VX,VY,VZ))`, `ecef` returning `state(ecef(X,Y,Z), ecef(VX,VY,VZ))`, and `wgs84_3d` returning `state(geographic(Latitude,Longitude,Height), enu(East,North,Up))`.

Compilation flags:

`static`

Template:

`propagate_state(TLE,Time,Frame,State)`

Mode and number of proofs:

`propagate_state(++compound,++nonvar,++atom,--compound)` - `one_or_error`

Exceptions:

`Frame` is not one of the supported reference frame identifiers:

`domain_error(tle_reference_frame,Frame)`

`Time` is not one of the supported time specifications with the required numeric components:

`domain_error(tle_time,Time)`

---

propagate_state/5

Propagates a parsed TLE using the requested propagation model and returns `state(Position,Velocity)` in the requested frame. Velocity is derived directly from the propagated orbital elements in ECI and analytically transformed to the requested frame. Supported models are `approximate`, `approximate_near_earth`, `approximate_deep_space`, and `two_body`.

Compilation flags:

`static`

Template:

`propagate_state(TLE,Time,Frame,Model,State)`

Mode and number of proofs:

`propagate_state(++compound,++nonvar,++atom,++atom,--compound)` - `one_or_error`

Exceptions:

`Frame` is not one of the supported reference frame identifiers:

`domain_error(tle_reference_frame,Frame)`

`Model` is not one of the supported propagation model identifiers:

`domain_error(tle_propagation_model,Model)`

`Time` is not one of the supported time specifications with the required numeric components:

`domain_error(tle_time,Time)`

---

ground_track/5

Samples the propagated sub-satellite ground track between two UTC `date_time/6` instants separated by a positive step size in seconds using the default `approximate` model, returning `sample(DateTime, geographic(Latitude,Longitude,Height))` terms.

Compilation flags:

`static`

Template:

`ground_track(TLE,StartDateTime,EndDateTime,StepSeconds,Samples)`

Mode and number of proofs:

`ground_track(++compound,++compound,++compound,++number,--list(compound))` - `one_or_error`

Exceptions:

`StartDateTime`, `EndDateTime`, or `StepSeconds` do not define a valid ground-track sampling interval for the default model:

`domain_error(tle_ground_track,[StartDateTime,EndDateTime,StepSeconds,approximate])`

---

ground_track/6

Samples the propagated sub-satellite ground track between two UTC `date_time/6` instants separated by a positive step size in seconds using the requested propagation model. Supported models are `approximate`, `approximate_near_earth`, `approximate_deep_space`, and `two_body`.

Compilation flags:

`static`

Template:

`ground_track(TLE,StartDateTime,EndDateTime,StepSeconds,Model,Samples)`

Mode and number of proofs:

`ground_track(++compound,++compound,++compound,++number,++atom,--list(compound))` - `one_or_error`

Exceptions:

`Model` is not one of the supported propagation model identifiers:

`domain_error(tle_propagation_model,Model)`

`StartDateTime`, `EndDateTime`, or `StepSeconds` do not define a valid ground-track sampling interval for the requested model:

`domain_error(tle_ground_track,[StartDateTime,EndDateTime,StepSeconds,Model])`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

tle_orbits, *crs_projections_protocol*

1.327 tokenization

object

1.327.1 `english_tokenizer`

English tokenization and sentence splitting provider using portable rule-based heuristics and a curated abbreviation inventory.

Availability:
`logtalk_load(tokenization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-05

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *tokenizer_language_protocol*
Imports:
`public` *tokenizer_rules*

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *split_sentence_codes/3* *tokenize_codes/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*

- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

tokenizer(Representation,Language), *tokenizer_rules*

protocol

1.327.2 `sentence_splitter_protocol`

Sentence splitting protocol for text represented as atoms, lists of characters, or lists of character codes.

Availability:
`logtalk_load(tokenization(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *split_sentences/2*
  - *split_sentences/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

split_sentences/2

Splits text into sentences using the default options.

Compilation flags:
`static`

Template:
`split_sentences(Text,Sentences)`

Mode and number of proofs:
`split_sentences(+text,-list(text))` - `one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Language` parameter is a variable:
`instantiation_error`
The `Language` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Language)`
The `Language` parameter is an object identifier but not an object conforming to `tokenizer_language_protocol`:
`domain_error(tokenizer_language_protocol,Language)`
`Text` is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but `Text` is not an atom:
`type_error(atom,Text)`
The `Representation` parameter is `chars` but `Text` is not a list of characters:
`type_error(chars,Text)`
The `Representation` parameter is `codes` but `Text` is not a list of character codes:
`type_error(codes,Text)`

split_sentences/3

Splits text into sentences using the given tokenization options.

Compilation flags:

`static`

Template:

`split_sentences(Text,Sentences,Options)`

Mode and number of proofs:

`split_sentences(+text,-list(text),+list(compound))` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Language` parameter is a variable:

`instantiation_error`

The `Language` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Language)`

The `Language` parameter is an object identifier but not an object conforming to `tokenizer_language_protocol`:

`domain_error(tokenizer_language_protocol,Language)`

`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:

`type_error(list,Options)`

An element `Option` of the list `Options` is a variable:

`instantiation_error`

An element `Option` of the list `Options` is neither a variable nor a compound term:

`type_error(compound,Option)`

An element `Option` of the list `Options` is a compound term but not a valid option:

`domain_error(option,Option)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

tokenizer_protocol, *tokenizer_language_protocol*, *tokenizer(Representation,Language)*

object

1.327.3 `tokenizer(Representation,Language)`

- `Representation` - Text representation. Valid values are `atom`, `codes`, and `chars`.
- `Language` - Object implementing the `tokenizer_language_protocol` protocol.

Tokenizer and sentence splitter parameterized by text representation and language provider.

Availability:
`logtalk_load(tokenization(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-10-06

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *tokenizer_protocol*
`public` *sentence_splitter_protocol*
Imports:
`public` *options*
Uses:
list
type

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *split_sentences/2* *split_sentences/3* *tokenize/2* *tokenize/3* *tokenize_sentences/2* *tokenize_sentences/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

tokenizer_protocol, *sentence_splitter_protocol*, *tokenizer_language_protocol*

protocol

1.327.4 `tokenizer_language_protocol`

Language provider protocol for tokenization and sentence splitting using canonical character-code lists.

Availability:
```
logtalk_load(tokenization(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-05

Compilation flags:
```
static
```

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *tokenize_codes/3*
  - *split_sentence_codes/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

tokenize_codes/3

Tokenizes a character-code list using validated, merged facade options.

Compilation flags:

```
static
```

Template:

```
tokenize_codes(Codes,Tokens,Options)
```

Mode and number of proofs:

```
tokenize_codes(+list(character_code),-list(list(character_code)),+list(compound))
- one
```

---

split_sentence_codes/3

Splits a character-code list into sentence character-code lists using validated, merged facade options.

Compilation flags:

```
static
```

Template:

```
split_sentence_codes(Codes,Sentences,Options)
```

Mode and number of proofs:

```
split_sentence_codes(+list(character_code),-list(list(character_code)),
+list(compound)) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

tokenizer_protocol, *sentence_splitter_protocol*, *tokenizer(Representation,Language)*

protocol

1.327.5 `tokenizer_protocol`

Tokenization protocol for text represented as atoms, lists of characters, or lists of character codes.

Availability:
`logtalk_load(tokenization(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-11-11

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - `tokenize/2`
  - `tokenize/3`
  - `tokenize_sentences/2`
  - `tokenize_sentences/3`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

tokenize/2

Tokenizes text using the default options.

Compilation flags:
`static`

Template:
`tokenize(Text,Tokens)`
Mode and number of proofs:
`tokenize(+text,-list(text))` - `one_or_error`

Exceptions:
The `Representation` parameter is a variable:
`instantiation_error`
The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:
`domain_error(text_representation,Representation)`
The `Language` parameter is a variable:
`instantiation_error`
The `Language` parameter neither a variable nor an object identifier:
`type_error(object_identifier,Language)`
The `Language` parameter is an object identifier but not an object conforming to `tokenizer_language_protocol`:
`domain_error(tokenizer_language_protocol,Language)`
`Text` is not a ground term:
`instantiation_error`
The `Representation` parameter is `atom` but `Text` is not an atom:
`type_error(atom,Text)`
The `Representation` parameter is `chars` but `Text` is not a list of characters:
`type_error(chars,Text)`
The `Representation` parameter is `codes` but `Text` is not a list of character codes:
`type_error(codes,Text)`

---

tokenize/3

Tokenizes text using the given options. Recognized options are `keep_punctuation(Boolean)`, `lowercase(Boolean)`, `normalize_quotes(Boolean)`, and `normalize_dashes(Boolean)`.

Compilation flags:
`static`

Template:
`tokenize(Text,Tokens,Options)`
Mode and number of proofs:
`tokenize(+text,-list(text),+list(compound))` - `one_or_error`

Exceptions:

The Representation parameter is a variable:
    instantiation_error
The Representation parameter is neither a variable nor atom, chars, or codes:
    domain_error(text_representation,Representation)
The Language parameter is a variable:
    instantiation_error
The Language parameter neither a variable nor an object identifier:
    type_error(object_identifier,Language)
The Language parameter is an object identifier but not an object conforming to tokenizer_language_protocol:
    domain_error(tokenizer_language_protocol,Language)
Text is not a ground term:
    instantiation_error
The Representation parameter is atom but Text is not an atom:
    type_error(atom,Text)
The Representation parameter is chars but Text is not a list of characters:
    type_error(chars,Text)
The Representation parameter is codes but Text is not a list of character codes:
    type_error(codes,Text)
Options is a variable:
    instantiation_error
Options is neither a variable nor a list:
    type_error(list,Options)
An element Option of the list Options is a variable:
    instantiation_error
An element Option of the list Options is neither a variable nor a compound term:
    type_error(compound,Option)
An element Option of the list Options is a compound term but not a valid option:
    domain_error(option,Option)

---

tokenize_sentences/2

Splits text into sentences and tokenizes each sentence using the default options.

Compilation flags:
    static

Template:
    tokenize_sentences(Text,TokenizedSentences)
Mode and number of proofs:
    tokenize_sentences(+text,-list(list(text))) - one_or_error

Exceptions:
The Representation parameter is a variable:
    instantiation_error
The Representation parameter is neither a variable nor atom, chars, or codes:
    domain_error(text_representation,Representation)
The Language parameter is a variable:

`instantiation_error`

The `Language` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Language)`

The `Language` parameter is an object identifier but not an object conforming to `tokenizer_language_protocol`:

`domain_error(tokenizer_language_protocol,Language)`

`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

`tokenize_sentences/3`

Splits text into sentences and tokenizes each sentence using the given options.

Compilation flags:

`static`

Template:

`tokenize_sentences(Text,TokenizedSentences,Options)`

Mode and number of proofs:

`tokenize_sentences(+text,-list(list(text)),+list(compound))` - `one_or_error`

Exceptions:

The `Representation` parameter is a variable:

`instantiation_error`

The `Representation` parameter is neither a variable nor `atom`, `chars`, or `codes`:

`domain_error(text_representation,Representation)`

The `Language` parameter is a variable:

`instantiation_error`

The `Language` parameter neither a variable nor an object identifier:

`type_error(object_identifier,Language)`

The `Language` parameter is an object identifier but not an object conforming to `tokenizer_language_protocol`:

`domain_error(tokenizer_language_protocol,Language)`

`Text` is not a ground term:

`instantiation_error`

The `Representation` parameter is `atom` but `Text` is not an atom:

`type_error(atom,Text)`

The `Representation` parameter is `chars` but `Text` is not a list of characters:

`type_error(chars,Text)`

The `Representation` parameter is `codes` but `Text` is not a list of character codes:

`type_error(codes,Text)`

`Options` is a variable:

`instantiation_error`

`Options` is neither a variable nor a list:
`type_error(list,Options)`
An element `Option` of the list `Options` is a variable:
`instantiation_error`
An element `Option` of the list `Options` is neither a variable nor a compound term:
`type_error(compound,Option)`
An element `Option` of the list `Options` is a compound term but not a valid option:
`domain_error(option,Option)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

sentence_splitter_protocol, *tokenizer_language_protocol*, *tokenizer(Representation,Language)*

category

1.327.6 `tokenizer_rules`

Reusable portable rules for tokenization and sentence splitting. Importing objects supply language-specific abbreviation and word punctuation hooks. URL and email address recognition delegates to the `url` library.

Availability:
`logtalk_load(tokenization(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-09-05

Compilation flags:
`static`

Implements:
`public` *tokenizer_language_protocol*
Extends:
`public` *options*
Uses:
list
number_grammars(Format)

string(Representation)
url(Representation)

Remarks:
(none)

Inherited public predicates:
check_option/1 *check_options/1* *default_option/1* *default_options/1* *option/2* *option/3* *split_sentence_codes/3* *tokenize_codes/3* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
  - `abbreviation/1`
  - `non_terminal_abbreviation/1`
  - `internal_apostrophe/1`
  - `internal_hyphen/1`
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

`abbreviation/1`

Enumerates canonical lowercase abbreviations, including their periods.

Compilation flags:
`static`

Template:
`abbreviation(Abbreviation)`
Mode and number of proofs:
`abbreviation(?atom)` - `zero_or_more`

`non_terminal_abbreviation/1`

Enumerates abbreviations that do not terminate a sentence when followed by another token.

Compilation flags:
`static`

Template:
`non_terminal_abbreviation(Abbreviation)`

Mode and number of proofs:

```
non_terminal_abbreviation(?atom) - zero_or_more
```

---

internal_apostrophe/1

Enumerates apostrophe character codes accepted inside words.

Compilation flags:

```
static
```

Template:

```
internal_apostrophe(Code)
```

Mode and number of proofs:

```
internal_apostrophe(?character_code) - zero_or_more
```

---

internal_hyphen/1

Enumerates hyphen character codes accepted inside words.

Compilation flags:

```
static
```

Template:

```
internal_hyphen(Code)
```

Mode and number of proofs:

```
internal_hyphen(?character_code) - zero_or_more
```

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

tokenizer_language_protocol, *english_tokenizer*, *url(Representation)*

1.328 toml

object

1.328.1 `toml`

TOML parser and generator. Uses compound terms for parsed TOML tables, dashes for parsed TOML pairs, and atoms for parsed TOML string values.

Availability:
`logtalk_load(toml(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-09

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *toml(compound,dash,atom)*

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.328.2 `toml(StringRepresentation)`

- `StringRepresentation` - Text representation to be used when decoding TOML string values. Possible values are `atom` (default), `chars`, and `codes`.

TOML parser and generator. Uses compound terms for parsed TOML tables and dashes for parsed TOML pairs.

Availability:
`logtalk_load(toml(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-09

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *toml(compound,dash,StringRepresentation)*

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.328.3 `toml(ObjectRepresentation,PairRepresentation,StringRepresentation)`

- `ObjectRepresentation` - Object representation to be used when decoding TOML tables. Possible values are `compound` (default) and `curly`.
- `PairRepresentation` - Pair representation to be used when decoding TOML tables. Possible values are `dash` (default), `equal`, and `colon`.
- `StringRepresentation` - Text representation to be used when decoding TOML basic and literal string values. Possible values are `atom` (default), `chars`, and `codes`.

TOML parser and generator.

Availability:
`logtalk_load(toml(loader))`

Author: Paulo Moura
Version: 1:1:2
Date: 2026-05-18

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *toml_protocol*
Uses:
list
reader

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.328.4 `toml_protocol`

TOML parser and generator protocol.

Availability:
`logtalk_load(toml(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-09

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses TOML content read from the given source (`file(Path)`, `stream(Stream)`, `codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) into a ground term representing the parsed TOML document.

Compilation flags:
`static`

Template:
`parse(Source,TOML)`
Mode and number of proofs:
`parse(++compound,--ground)` - `zero_or_one`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(toml_source,Source)`

---

generate/2

Generates TOML output using the representation specified in the first argument (`file(Path)`, `stream(Stream)`, `codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) from the ground TOML term in the second argument.

Compilation flags:
`static`

Template:
`generate(Sink,TOML)`
Mode and number of proofs:
`generate(++compound,+ground)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`TOML` is a variable:
`instantiation_error`
`TOML` is not a valid TOML term:
`domain_error(toml_term,TOML)`
`Sink` cannot be generated:
`domain_error(toml_sink,Sink)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.329 tool_diagnostics

category

1.329.1 `tool_diagnostics_common`

Default definitions for diagnostics targets, rule enumeration, and summary/breakdown helpers.

Availability:
`logtalk_load(tool_diagnostics(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-01

Compilation flags:
`static`

Implements:
`public` *tool_diagnostics_protocol*
Uses:
list

Remarks:
(none)

Inherited public predicates:
diagnostic/2 *diagnostic/3* *diagnostic_rule/5* *diagnostic_rules/1* *diagnostic_target/1* *diagnostics/2* *diagnostics/3* *diagnostics_preflight/2* *diagnostics_preflight/3* *diagnostics_summary/2* *diagnostics_summary/3* *diagnostics_tool/5*

- *Public predicates*
- *Protected predicates*
  - *context_summaries/2*
  - *diagnostics_breakdown/2*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

context_summaries/2

Returns a list of context summary terms for a list of diagnostics.

Compilation flags:
`static`

Template:
`context_summaries(Diagnostics,ContextSummaries)`
Mode and number of proofs:
`context_summaries(+list(compound),-list(compound))` - `one`

---

diagnostics_breakdown/2

Returns a breakdown term with rule, severity, and confidence counts for a list of diagnostics.

Compilation flags:
`static`

Template:
`diagnostics_breakdown(Diagnostics,Breakdown)`
Mode and number of proofs:
`diagnostics_breakdown(+list(compound),-compound)` - `one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.329.2 tool_diagnostics_protocol

Common machine-readable diagnostics protocol for developer tools.

Availability:
`logtalk_load(tool_diagnostics(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-03-31

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

- Targets and options: Implementing objects should enumerate supported targets using `diagnostic_target/1` and predicates taking an `Options` argument must accept `explanations(Boolean)`.
- Tool metadata predicate: Tool metadata is exposed using `diagnostics_tool/5`.
- Rule descriptor predicate: Rule descriptors are exposed using `diagnostic_rule/5` and collected in stable order using `diagnostic_rules/1`.
- Diagnostic and preflight terms: Findings and analysis prerequisites are represented using `diagnostic/8` and `preflight_issue/7` terms.
- Summary term: Summaries are represented using `diagnostics_summary/5` terms with supporting breakdown and count terms.

Inherited public predicates:
(none)

- *Public predicates*
  - *diagnostics_tool/5*
  - *diagnostic_target/1*
  - *diagnostic_rule/5*
  - *diagnostic_rules/1*
  - *diagnostic/3*
  - *diagnostic/2*
  - *diagnostics/3*
  - *diagnostics/2*
  - *diagnostics_summary/3*
  - *diagnostics_summary/2*
  - *diagnostics_preflight/3*
  - *diagnostics_preflight/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

diagnostics_tool/5

Returns tool metadata.

Compilation flags:
`static`

Template:
`diagnostics_tool(Id,Name,Version,InformationURI,Properties)`
Mode and number of proofs:
`diagnostics_tool(?atom,?atom,?atom,?atom,?list(compound))` - `zero_or_one`

---

diagnostic_target/1

Enumerates supported diagnostics target patterns.

Compilation flags:
`static`

Template:
`diagnostic_target(Target)`
Mode and number of proofs:
`diagnostic_target(?nonvar)` - `zero_or_more`

---

diagnostic_rule/5

Enumerates diagnostic rule descriptors.

Compilation flags:
`static`

Template:
`diagnostic_rule(RuleId,ShortDescription,FullDescription,DefaultSeverity,`
`Properties)`
Mode and number of proofs:
`diagnostic_rule(?atom,?atom,?atom,?atom,?list(compound))` - `zero_or_more`

---

diagnostic_rules/1

Returns all supported diagnostic rule descriptors in a stable order.

Compilation flags:
`static`

Template:
`diagnostic_rules(Rules)`
Mode and number of proofs:
`diagnostic_rules(-list(compound))` - `one`

---

diagnostic/3

Enumerates, by backtracking, diagnostics for a target using the given options. Diagnostics are returned using terms of the form `diagnostic(RuleId, Severity, Confidence, Message, Context, File, Lines, Properties)`. All implementations must accept the common option `explanations(Boolean)`.

Compilation flags:
`static`

Template:
`diagnostic(Target,Diagnostic,Options)`
Mode and number of proofs:
`diagnostic(+nonvar,-compound,+list(compound))` - `zero_or_more`

---

diagnostic/2

Enumerates, by backtracking, diagnostics for a target using default options. Diagnostics are returned using terms of the form `diagnostic(RuleId, Severity, Confidence, Message, Context, File, Lines, Properties)`.

Compilation flags:
`static`

Template:
`diagnostic(Target,Diagnostic)`
Mode and number of proofs:
`diagnostic(+nonvar,-compound)` - `zero_or_more`

---

diagnostics/3

Returns an ordered set of diagnostics for a target using the given options. Diagnostics are returned using terms of the form `diagnostic(RuleId, Severity, Confidence, Message, Context, File, Lines, Properties)`. All implementations must accept the common option `explanations(Boolean)`.

Compilation flags:
`static`

Template:
`diagnostics(Target,Diagnostics,Options)`
Mode and number of proofs:
`diagnostics(+nonvar,-list(compound),+list(compound))` - `one`

---

diagnostics/2

Returns an ordered set of diagnostics for a target using default options. Diagnostics are returned using terms of the form `diagnostic(RuleId, Severity, Confidence, Message, Context, File, Lines, Properties)`.

Compilation flags:
`static`

Template:
`diagnostics(Target,Diagnostics)`
Mode and number of proofs:
`diagnostics(+nonvar,-list(compound))` - `one`

---

diagnostics_summary/3

Returns a machine-readable summary for a target using the given options. The summary counts diagnostics only and does not include preflight issues. All implementations must accept the common option `explanations(Boolean)`.

Compilation flags:
`static`

Template:
`diagnostics_summary(Target,Summary,Options)`
Mode and number of proofs:
`diagnostics_summary(+nonvar,-compound,+list(compound))` - `one`

---

diagnostics_summary/2

Returns a machine-readable summary for a target using default options. The summary counts diagnostics only and does not include preflight issues.

Compilation flags:

```
static
```

Template:

```
diagnostics_summary(Target,Summary)
```

Mode and number of proofs:

```
diagnostics_summary(+nonvar,-compound) - one
```

---

diagnostics_preflight/3

Returns an ordered set of machine-readable preflight issues for a target using the given options. Preflight issues are returned using terms of the form `preflight_issue(Id, Severity, Message, Context, File, Lines, Properties)`. All implementations must accept the common option `explanations(Boolean)`.

Compilation flags:

```
static
```

Template:

```
diagnostics_preflight(Target,Issues,Options)
```

Mode and number of proofs:

```
diagnostics_preflight(+nonvar,-list(compound),+list(compound)) - one
```

---

diagnostics_preflight/2

Returns an ordered set of machine-readable preflight issues for a target using default options. Preflight issues are returned using terms of the form `preflight_issue(Id, Severity, Message, Context, File, Lines, Properties)`.

Compilation flags:

```
static
```

Template:

```
diagnostics_preflight(Target,Issues)
```

Mode and number of proofs:

```
diagnostics_preflight(+nonvar,-list(compound)) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.330 toon

object

1.330.1 `toon`

TOON parser and generator. Uses curly terms for parsed TOON objects, dashes for parsed TOON pairs, and atoms for parsed TOON strings.

Availability:
`logtalk_load(toon(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2025-12-07

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *toon(curly,dash,atom)*

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.330.2 `toon(StringRepresentation)`

- `StringRepresentation` - Text representation to be used when decoding TOON strings. Possible values are `atom` (default), `chars`, and `codes`.

TOON parser and generator. Uses curly terms for parsed TOON objects and dashes for parsed TOON pairs.

Availability:
`logtalk_load(toon(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2025-12-07

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *toon(curly,dash,StringRepresentation)*

Remarks:
(none)

Inherited public predicates:
generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.330.3 `toon(ObjectRepresentation,PairRepresentation,StringRepresentation)`

- `ObjectRepresentation` - Object representation to be used when decoding TOON objects. Possible values are `curly` (default) and `list`.
- `PairRepresentation` - Pair representation to be used when decoding TOON objects. Possible values are `dash` (default), `equal`, and `colon`.
- `StringRepresentation` - Text representation to be used when decoding TOON strings. Possible values are `atom` (default), `chars`, and `codes`.

TOON (Token-Oriented Object Notation) parser and generator. TOON is a compact, human-readable, line-oriented format that encodes the JSON data model.

Availability:

`logtalk_load(toon(loader))`

Author: Paulo Moura
Version: 0:2:0
Date: 2026-04-17

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *toon_protocol*

Uses:

list
meta
reader

Remarks:

(none)

Inherited public predicates:

generate/2 *parse/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.330.4 `toon_protocol`

TOON (Token-Oriented Object Notation) parser and generator protocol.

Availability:

```
logtalk_load(toon(loader))
```

Author: Paulo Moura
Version: 0:1:0
Date: 2025-12-15

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*

  - *parse/2*
  - *generate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses the TOON contents read from the given source (`codes(List)`, `stream(Stream)`, `file(Path)`, `chars(List)`, or `atom(Atom)`) into a term. Throws an error if the TOON contents cannot be parsed.

Compilation flags:
`static`

Template:
`parse(Source,Term)`
Mode and number of proofs:
`parse(++compound,--term)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(toon_source,Source)`

---

generate/2

Generates the content using the representation specified in the first argument (`codes(List)`, `stream(Stream)`, `file(Path)`, `chars(List)`, or `atom(Atom)`) for the term in the second argument. Throws an error if this term cannot be processed.

Compilation flags:
`static`

Template:
`generate(Sink,Term)`
Mode and number of proofs:
`generate(+compound,++term)` - `one_or_error`

Exceptions:
`Sink` is a variable:
`instantiation_error`
`Sink` is neither a variable nor a valid sink:
`domain_error(toon_sink,Sink)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.331 toychr

object

1.331.1 `toychrdb`

Simple CHR interpreter/debugger based on the refined operational semantics of CHRs.

Availability:
`logtalk_load(toychr(loader))`

Author: Gregory J. Duck; adapted to Logtalk by Paulo Moura.
Version: 0:8:0
Date: 2026-08-10

Copyright: Copright 2004 Gregory J. Duck; Copyright 2019-2024 Paulo Moura
License: GPL-2.0-or-later

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Uses:
list
user

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*

    - *chr_is/2*
    - *chr_trace/0*
    - *chr_notrace/0*
    - *chr_spy/1*
    - *chr_nospy/0*
    - *chr_no_spy/1*
    - *chr_option/2*
  - *Protected predicates*
    - *current_prog/1*
    - *chr_option_print_trace/0*
    - *chr_option_trace_interactive/0*
    - *chr_option_optimization_level/1*
    - *chr_option_show_stack/0*
    - *chr_option_show_store/0*
    - *chr_option_show_history/0*
    - *chr_option_show_id/0*
    - *chr_option_allow_deep_guards/0*
    - *chr_next_state/1*
    - *chr_spy_point/1*
  - *Private predicates*
    - *chr_rule_/1*
  - *Operators*

Public predicates

chr_is/2

Runs a goal against the entity CHR program and returns the resulting constraint store as a conjunction.

Compilation flags:

```
static
```

Template:

```
chr_is(Result,Goal)
```

Mode and number of proofs:

```
chr_is(?nonvar,+callable) - zero_or_one
```

chr_trace/0

Enables the CHR interactive trace debugger. Similar to the trace/0 standard predicate.

Compilation flags:
`static`

Mode and number of proofs:
`chr_trace` - `one`

---

chr_notrace/0

Disables the CHR interactive trace debugger. Similar to the notrace/0 standard predicate.

Compilation flags:
`static`

Mode and number of proofs:
`chr_notrace` - `one`

---

chr_spy/1

Sets a spy point on the activation of constraints matching the given pattern.

Compilation flags:
`static`

Template:
`chr_spy(Pattern)`
Mode and number of proofs:
`chr_spy(@callable)` - `one`

---

chr_nospy/0

Disables all spy points.

Compilation flags:
`static`

Mode and number of proofs:
`chr_nospy` - `one`

---

chr_no_spy/1

Not yet implemented; disables the spy point on constraints matching the given pattern.

Compilation flags:
`static`

Template:
`chr_no_spy(Pattern)`
Mode and number of proofs:
`chr_no_spy(@callable)` - `zero`

---

chr_option/2

Sets a debugger or compiler option. See the file documentation for the supported options and their values.

Compilation flags:
`static`

Template:
`chr_option(Option,Value)`
Mode and number of proofs:
`chr_option(+atom,+atom)` - `zero_or_one`

Protected predicates

current_prog/1

Compiled CHR program generated from the entity chr_constraint, handler, and rules terms.

Compilation flags:
`static`

Template:
`current_prog(Program)`
Mode and number of proofs:
`current_prog(?callable)` - `zero_or_one`

---

chr_option_print_trace/0

True when trace printing is enabled.

Compilation flags:
`dynamic`

Mode and number of proofs:
`chr_option_print_trace` - `zero_or_one`

---

chr_option_trace_interactive/0

True when interactive tracing is enabled.

Compilation flags:
`dynamic`

Mode and number of proofs:
`chr_option_trace_interactive` - `zero_or_one`

---

chr_option_optimization_level/1

Current compiler optimization level.

Compilation flags:
`dynamic`

Template:
`chr_option_optimization_level(Level)`
Mode and number of proofs:
`chr_option_optimization_level(?integer)` - `zero_or_one`

---

chr_option_show_stack/0

True when the execution stack is shown while tracing.

Compilation flags:
`dynamic`

Mode and number of proofs:
`chr_option_show_stack` - `zero_or_one`

---

chr_option_show_store/0

True when the constraint store is shown while tracing.

Compilation flags:
dynamic

Mode and number of proofs:
chr_option_show_store - zero_or_one

---

chr_option_show_history/0

True when the propagation history is shown while tracing.

Compilation flags:
dynamic

Mode and number of proofs:
chr_option_show_history - zero_or_one

---

chr_option_show_id/0

True when the current constraint identifier is shown while tracing.

Compilation flags:
dynamic

Mode and number of proofs:
chr_option_show_id - zero_or_one

---

chr_option_allow_deep_guards/0

True when CHR constraints are allowed in rule guards or if-conditions.

Compilation flags:
dynamic

Mode and number of proofs:
chr_option_allow_deep_guards - zero_or_one

---

chr_next_state/1

Stack length recorded to detect when the next debugger step is reached.

Compilation flags:
`dynamic`

Template:
`chr_next_state(Length)`
Mode and number of proofs:
`chr_next_state(?integer)` - `zero_or_one`

---

chr_spy_point/1

Asserted clause matching constraints on which a spy point is set.

Compilation flags:
`dynamic`

Template:
`chr_spy_point(Constraint)`
Mode and number of proofs:
`chr_spy_point(@callable)` - `zero_or_more`

Private predicates

chr_rule_/1

Asserted CHR rule term collected during term-expansion, consumed when compiling the entity program.

Compilation flags:
`dynamic`

Template:
`chr_rule_(Rule)`
Mode and number of proofs:
`chr_rule_(?callable)` - `zero_or_more`

Operators

(none)

1.332 tries

object

1.332.1 `trie(Representation)`

- `Representation` - String representation. Valid values are `atom`, `codes`, and `chars`.

Persistent trie predicates supporting different string representations.

Availability:
`logtalk_load(tries(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-05

Compilation flags:
`static, context_switching_calls`

Uses:
list
pairs
string(Representation)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *new/1*
  - *as_trie/2*
  - *as_list/2*
  - *as_list/3*
  - *strings/2*
  - *strings/3*
  - *values/2*
  - *values/3*
  - *empty/1*
  - *insert/4*
  - *lookup/3*
  - *update/4*

      - *update/5*
      - *delete/4*
      - *lookup_prefix/4*
      - *prefixes/3*
      - *longest_prefix/4*
      - *delete_prefix/3*
      - *size/2*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

new/1

Creates an empty trie.

Compilation flags:
```
static
```

Template:
```
new(Trie)
```
Mode and number of proofs:
```
new(-trie) - one
```

---

as_trie/2

Converts a list of string-value pairs to a trie. When a string occurs multiple times, the last value is retained.

Compilation flags:
```
static
```

Template:
```
as_trie(Pairs,Trie)
```
Mode and number of proofs:
```
as_trie(@list(pairs),-trie) - one
```

---

as_list/2

Converts a trie to a lexicographically ordered list of string-value pairs.

Compilation flags:
```
static
```

Template:
```
as_list(Trie,Pairs)
```
Mode and number of proofs:
```
as_list(@trie,-list(pairs)) - one
```

---

as_list/3

Returns a lexicographically ordered list of all string-value pairs whose strings start with the given prefix. Fails if no string starts with the prefix.

Compilation flags:
```
static
```

Template:
```
as_list(Trie,Prefix,Pairs)
```
Mode and number of proofs:
```
as_list(@trie,+text,-list(pairs)) - zero_or_one
```

---

strings/2

Returns all strings in a trie in lexicographic order.

Compilation flags:
```
static
```

Template:
```
strings(Trie,Strings)
```
Mode and number of proofs:
```
strings(@trie,-list(text)) - one
```

---

strings/3

Returns all strings that start with the given prefix in lexicographic order. Fails if no string starts with the prefix.

Compilation flags:

```
static
```

Template:

```
strings(Trie,Prefix,Strings)
```

Mode and number of proofs:

```
strings(@trie,+text,-list(text)) - zero_or_one
```

---

values/2

Returns all values in lexicographic order of their associated strings.

Compilation flags:

```
static
```

Template:

```
values(Trie,Values)
```

Mode and number of proofs:

```
values(@trie,-list) - one
```

---

values/3

Returns all values associated with strings that start with the given prefix, in lexicographic order of the strings. Fails if no string starts with the prefix.

Compilation flags:

```
static
```

Template:

```
values(Trie,Prefix,Values)
```

Mode and number of proofs:

```
values(@trie,+text,-list) - zero_or_one
```

---

empty/1

True iff the trie is empty.

Compilation flags:
```
static
```

Template:
```
empty(Trie)
```
Mode and number of proofs:
```
empty(@trie) - zero_or_one
```

---

insert/4

Inserts a string-value pair into a trie, returning the updated trie. When the string already exists, its associated value is updated.

Compilation flags:
```
static
```

Template:
```
insert(OldTrie,String,Value,NewTrie)
```
Mode and number of proofs:
```
insert(+trie,+text,@term,-trie) - one
```

---

lookup/3

Looks up a matching string-value pair. With an unbound string, enumerates all pairs in lexicographic order.

Compilation flags:
```
static
```

Template:
```
lookup(String,Value,Trie)
```
Mode and number of proofs:
```
lookup(+text,?term,@trie) - zero_or_one
lookup(-text,?term,@trie) - zero_or_more
```

---

update/4

Updates the value associated with a string, returning the updated trie. Fails if the string is not stored.

Compilation flags:

```
static
```

Template:

```
update(OldTrie,String,NewValue,NewTrie)
```

Mode and number of proofs:

```
update(+trie,+text,+term,-trie) - zero_or_one
```

---

update/5

Updates the value associated with a string, returning the updated trie. Fails if the string is not stored or its value does not unify with the expected old value.

Compilation flags:

```
static
```

Template:

```
update(OldTrie,String,OldValue,NewValue,NewTrie)
```

Mode and number of proofs:

```
update(+trie,+text,?term,+term,-trie) - zero_or_one
```

---

delete/4

Deletes a matching string-value pair, returning the updated trie. Descendant strings are preserved.

Compilation flags:

```
static
```

Template:

```
delete(OldTrie,String,Value,NewTrie)
```

Mode and number of proofs:

```
delete(+trie,+text,?term,-trie) - zero_or_one
```

---

lookup_prefix/4

Enumerates all string-value pairs whose strings start with the given prefix, in lexicographic order.

Compilation flags:
`static`

Template:
`lookup_prefix(Prefix,String,Value,Trie)`
Mode and number of proofs:
`lookup_prefix(+text,-text,?term,@trie)` - `zero_or_more`

---

prefixes/3

Returns all stored string-value pairs whose strings are prefixes of the query, ordered from shortest to longest.

Compilation flags:
`static`

Template:
`prefixes(Trie,Query,Pairs)`
Mode and number of proofs:
`prefixes(@trie,+text,-list(pairs))` - `one`

---

longest_prefix/4

Returns the longest stored string that is a prefix of the query and its associated value. Fails if no stored string is a prefix.

Compilation flags:
`static`

Template:
`longest_prefix(Trie,Query,Prefix,Value)`
Mode and number of proofs:
`longest_prefix(@trie,+text,-text,-term)` - `zero_or_one`

---

delete_prefix/3

Deletes all string-value pairs whose strings start with the given prefix, returning the updated trie. Fails if no string starts with the prefix.

Compilation flags:
`static`

Template:
`delete_prefix(OldTrie,Prefix,NewTrie)`
Mode and number of proofs:
`delete_prefix(+trie,+text,-trie)` - `zero_or_one`

---

size/2

Number of string-value pairs stored in a trie.

Compilation flags:
`static`

Template:
`size(Trie,Size)`
Mode and number of proofs:
`size(@trie,?integer)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.333 trueskill_ranker

object

1.333.1 `trueskill_ranker`

TrueSkill ranker for ordered multiplayer matches with ranked teams, draws, and weighted player participation.

Availability:

```
logtalk_load(trueskill_ranker(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-11

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public ranking_dataset_common
public score_ranker_model_common
```

Uses:

avltree
list
univariate_distributions(Random)

Remarks:

(none)

Inherited public predicates:

check_option/1 *check_options/1* *check_ranker/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *grouped_dataset_groups/2* *grouped_dataset_items/2* *grouped_dataset_items/3* *grouped_dataset_summary/2* *learn/2* *learn/3* *multiplayer_dataset_connected_components/2* *multiplayer_dataset_declared_items/2* *multiplayer_dataset_declared_matches/2* *multiplayer_dataset_events/2* *multiplayer_dataset_items/2* *multiplayer_dataset_match_teams/3* *multiplayer_dataset_matches/2* *multiplayer_dataset_summary/2* *option/2* *option/3* *pairwise_dataset_connected_components/2* *pairwise_dataset_declared_items/2* *pairwise_dataset_items/2* *pairwise_dataset_matchups/2* *pairwise_dataset_preferences/2* *pairwise_dataset_summary/2* *pairwise_dataset_win_totals/2* *pairwise_measurement_dataset_connected_components/2* *pairwise_measurement_dataset_declared_items/2* *pairwise_measurement_dataset_items/2* *pairwise_measurement_dataset_measurements/2* *pairwise_measurement_dataset_summary/2* *print_ranker/1* *rank/3* *ranker_options/2* *scores/2* *temporal_pairwise_dataset_connected_components/2* *temporal_pairwise_dataset_declared_items/2* *temporal_pairwise_dataset_games/2* *temporal_pairwise_dataset_games/3* *temporal_pairwise_dataset_items/2* *temporal_pairwise_dataset_periods/2* *temporal_pairwise_dataset_summary/2* *valid_option/1* *valid_options/1* *valid_ranker/1* *validate_grouped_dataset/1* *validate_grouped_dataset/2* *validate_multiplayer_dataset/1* *validate_multiplayer_dataset/2* *validate_pairwise_dataset/1* *validate_pairwise_dataset/2* *validate_pairwise_measurement_dataset/1* *validate_pairwise_measurement_dataset/2* *validate_temporal_pairwise_dataset/1* *validate_temporal_pairwise_dataset/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

multiplayer_ranking_dataset_protocol, *ranker_protocol*, *glicko2_periodic_ranker*

1.334 truncated_svd_projection

object

1.334.1 `truncated_svd_projection`

Truncated singular value decomposition reducer for continuous datasets using a portable two-sided power-iteration solver over the data matrix.

Availability:

```
logtalk_load(truncated_svd_projection(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-07

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *dimension_reducer_common*

Uses:

format
linear_algebra
list
numberlist
pairs
population
type

Remarks:

(none)

Inherited public predicates:

check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
  - *learn/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

learn/3

Learns a truncated singular value decomposition reducer from the given dataset object using the specified options.

Compilation flags:

```
static
```

Template:

```
learn(Dataset,DimensionReducer,Options)
```

Mode and number of proofs:

```
learn(+object_identifier,-compound,+list(compound)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

pca_projection, *random_projection*

1.335 tsne_projection

object

1.335.1 `tsne_projection`

Exact t-distributed Stochastic Neighbor Embedding dimension reducer for continuous datasets.

Availability:
`logtalk_load(tsne_projection(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2026-10-21

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *dimension_reducer_common*
Provides:
logtalk::message_tokens//2
Uses:
avltree
fast_random(Algorithm)
format
linear_algebra
list
logtalk
pairs
population
type
user

Remarks:
(none)

Inherited public predicates:
check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - *gradient_matrix/6*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`gradient_matrix/6`

Computes the unfused training gradient matrix as a utility for tests and benchmarks.

Compilation flags:
`static`

Template:
`gradient_matrix(Probabilities,Weights,WeightSum,Embedding,ProbabilityScale,Gradients)`

Mode and number of proofs:
`gradient_matrix(+list(list(number)),+list(list(number)),+number,+list(list(number)),+number,-list(list(number)))` - `one`

Operators

(none)

See also

kernel_pca_projection, *pca_projection*, *random_projection*

1.336 tsv

object

1.336.1 `tsv`

TSV files reading and writing predicates using the option `Header-keep`.

Availability:
`logtalk_load(tsv(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2023-11-15

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *tsv(keep,false)*

Remarks:
(none)

Inherited public predicates:
read_file/2 *read_file/3* *read_file_by_line/2* *read_file_by_line/3* *read_stream/2* *read_stream/3* *read_stream_by_line/2* *read_stream_by_line/3* *write_file/3* *write_stream/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.336.2 `tsv(Header)`

- `Header` - Header handling option with possible values `skip` and `keep` (default).

Backward-compatible parametric object equivalent to using `tsv(_Header_, false)`.

Availability:
`logtalk_load(tsv(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-02-27

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *tsv(Header,false)*

Remarks:
(none)

Inherited public predicates:
read_file/2 *read_file/3* *read_file_by_line/2* *read_file_by_line/3* *read_stream/2* *read_stream/3* *read_stream_by_line/2* *read_stream_by_line/3* *write_file/3* *write_stream/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.336.3 `tsv(Header,Comments)`

- `Header` - Header handling option with possible values `skip` and `keep` (default).
- `Comments` - Comment handling option with possible values `true` and `false` (default). When `true`, lines starting with `#` are skipped when reading TSV files and streams.

TSV file and stream reading and writing predicates.

Availability:
`logtalk_load(tsv(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-02-25

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *tsv_protocol*
Uses:

list
logtalk
reader
type

Remarks:
(none)

Inherited public predicates:
read_file/2 *read_file/3* *read_file_by_line/2* *read_file_by_line/3* *read_stream/2* *read_stream/3* *read_stream_by_line/2* *read_stream_by_line/3* *write_file/3* *write_stream/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.336.4 `tsv_protocol`

TSV file and stream reading and writing protocol.

Availability:
`logtalk_load(tsv(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2025-05-07

Compilation flags:
`static`

Dependencies:

(none)

Remarks:

- Type-checking: Some of the predicate file and stream argument type-checking exceptions depend on the Prolog backend compliance with standards.

Inherited public predicates:
(none)

- *Public predicates*
  - *read_file/3*
  - *read_stream/3*
  - *read_file/2*
  - *read_stream/2*
  - *read_file_by_line/3*
  - *read_stream_by_line/3*
  - *read_file_by_line/2*
  - *read_stream_by_line/2*
  - *write_file/3*
  - *write_stream/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

read_file/3

Reads a TSV file saving the data as clauses for the specified object predicate. Fails if the file cannot be parsed.

Compilation flags:
`static`

Template:
`read_file(File,Object,Predicate)`

Mode and number of proofs:
`read_file(+atom,+object_identifier,+predicate_indicator)` - `zero_or_one`

Exceptions:
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`File` is an atom but not an existing file:
`existence_error(file,File)`
`File` is an existing file but cannot be opened for reading:
`permission_error(open,source_sink,File)`

`Object` is a variable:
`instantiation_error`
`Object` is neither a variable nor an object identifier:
`type_error(object_identifier,Object)`
`Object` is a valid object identifier but not an existing object:
`existence_error(object,Object)`
`Predicate` is a variable:
`instantiation_error`
`Predicate` is neither a variable nor a predicate indicator:
`type_error(predicate_indicator,Predicate)`
`Predicate` is a valid predicate indicator but not an existing public predicate:
`existence_error(predicate,Predicate)`

`read_stream/3`

Reads a TSV stream saving the data as clauses for the specified object predicate. Fails if the stream cannot be parsed.

Compilation flags:
`static`

Template:
`read_stream(Stream,Object,Predicate)`
Mode and number of proofs:
`read_stream(+stream_or_alias,+object_identifier,+predicate_indicator)` - `zero_or_one`

Exceptions:
`Stream` is a variable:
`instantiation_error`
`Stream` is neither a variable nor a stream-term or alias:
`domain_error(stream_or_alias,Stream)`
`Stream` is not an open stream:
`existence_error(stream,Stream)`
`Stream` is an output stream:
`permission_error(input,stream,Stream)`
`Stream` is a binary stream:
`permission_error(input,binary_stream,Stream)`
`Object` is a variable:
`instantiation_error`
`Object` is neither a variable nor an object identifier:
`type_error(object_identifier,Object)`
`Object` is a valid object identifier but not an existing object:
`existence_error(object,Object)`
`Predicate` is a variable:
`instantiation_error`
`Predicate` is neither a variable nor a predicate indicator:
`type_error(predicate_indicator,Predicate)`

`Predicate` is a valid predicate indicator but not an existing public predicate:
`existence_error(predicate,Predicate)`

---

read_file/2

Reads a TSV file returning the data as a list of rows, each row a list of fields. Fails if the file cannot be parsed.

Compilation flags:
`static`

Template:
`read_file(File,Rows)`
Mode and number of proofs:
`read_file(+atom,-list(list))` - `zero_or_one`

Exceptions:
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`File` is an atom but not an existing file:
`existence_error(file,File)`
`File` is an existing file but cannot be opened for reading:
`permission_error(open,source_sink,File)`

---

read_stream/2

Reads a TSV stream returning the data as a list of rows, each row a list of fields. Fails if the stream cannot be parsed.

Compilation flags:
`static`

Template:
`read_stream(Stream,Rows)`
Mode and number of proofs:
`read_stream(+stream_or_alias,-list(list))` - `zero_or_one`

Exceptions:
`Stream` is a variable:
`instantiation_error`
`Stream` is neither a variable nor a stream-term or alias:
`domain_error(stream_or_alias,Stream)`
`Stream` is not an open stream:
`existence_error(stream,Stream)`

`Stream` is an output stream:
```
permission_error(input,stream,Stream)
```
`Stream` is a binary stream:
```
permission_error(input,binary_stream,Stream)
```

`read_file_by_line/3`

Reads a TSV file saving the data as clauses for the specified object predicate. The file is read line by line. Fails if the file cannot be parsed.

Compilation flags:
```
static
```

Template:
```
read_file_by_line(File,Object,Predicate)
```
Mode and number of proofs:
```
read_file_by_line(+atom,+object_identifier,+predicate_indicator) - zero_or_one
```

Exceptions:
`File` is a variable:
```
instantiation_error
```
`File` is neither a variable nor an atom:
```
type_error(atom,File)
```
`File` is an atom but not an existing file:
```
existence_error(file,File)
```
`File` is an existing file but cannot be opened for reading:
```
permission_error(open,source_sink,File)
```
`Object` is a variable:
```
instantiation_error
```
`Object` is neither a variable nor an object identifier:
```
type_error(object_identifier,Object)
```
`Object` is a valid object identifier but not an existing object:
```
existence_error(object,Object)
```
`Predicate` is a variable:
```
instantiation_error
```
`Predicate` is neither a variable nor a predicate indicator:
```
type_error(predicate_indicator,Predicate)
```
`Predicate` is a valid predicate indicator but not an existing public predicate:
```
existence_error(predicate,Predicate)
```

---

read_stream_by_line/3

Reads a TSV stream saving the data as clauses for the specified object predicate. The stream is read line by line. Fails if the stream cannot be parsed.

Compilation flags:

`static`

Template:

`read_stream_by_line(Stream,Object,Predicate)`

Mode and number of proofs:

`read_stream_by_line(+stream_or_alias,+object_identifier,+predicate_indicator)` - `zero_or_one`

Exceptions:

`Stream` is a variable:

`instantiation_error`

`Stream` is neither a variable nor a stream-term or alias:

`domain_error(stream_or_alias,Stream)`

`Stream` is not an open stream:

`existence_error(stream,Stream)`

`Stream` is an output stream:

`permission_error(input,stream,Stream)`

`Stream` is a binary stream:

`permission_error(input,binary_stream,Stream)`

`Object` is a variable:

`instantiation_error`

`Object` is neither a variable nor an object identifier:

`type_error(object_identifier,Object)`

`Object` is a valid object identifier but not an existing object:

`existence_error(object,Object)`

`Predicate` is a variable:

`instantiation_error`

`Predicate` is neither a variable nor a predicate indicator:

`type_error(predicate_indicator,Predicate)`

`Predicate` is a valid predicate indicator but not an existing public predicate:

`existence_error(predicate,Predicate)`

---

read_file_by_line/2

Reads a TSV file returning the data as a list of rows, each row a list of fields. The file is read line by line. Fails if the file cannot be parsed.

Compilation flags:

`static`

Template:

`read_file_by_line(File,Rows)`

Mode and number of proofs:

`read_file_by_line(+atom,-list(list))` - `zero_or_one`

Exceptions:

`File` is a variable:

`instantiation_error`

`File` is neither a variable nor an atom:

`type_error(atom,File)`

`File` is an atom but not an existing file:

`existence_error(file,File)`

`File` is an existing file but cannot be opened for reading:

`permission_error(open,source_sink,File)`

`read_stream_by_line/2`

Reads a TSV stream returning the data as a list of rows, each row a list of fields. The stream is read line by line. Fails if the stream cannot be parsed.

Compilation flags:

`static`

Template:

`read_stream_by_line(Stream,Rows)`

Mode and number of proofs:

`read_stream_by_line(+stream_or_alias,-list(list))` - `zero_or_one`

Exceptions:

`Stream` is a variable:

`instantiation_error`

`Stream` is neither a variable nor a stream-term or alias:

`domain_error(stream_or_alias,Stream)`

`Stream` is not an open stream:

`existence_error(stream,Stream)`

`Stream` is an output stream:

`permission_error(input,stream,Stream)`

`Stream` is a binary stream:

`permission_error(input,binary_stream,Stream)`

---

write_file/3

Writes a TSV file with the data represented by the solutions to the specified object predicate.

Compilation flags:
`static`

Template:
`write_file(File,Object,Predicate)`
Mode and number of proofs:
`write_file(+atom,+object_identifier,+predicate_indicator)` - `one`

Exceptions:
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor an atom:
`type_error(atom,File)`
`File` is an atom but cannot be opened for writing:
`permission_error(open,source_sink,File)`
`Object` is a variable:
`instantiation_error`
`Object` is neither a variable nor an object identifier:
`type_error(object_identifier,Object)`
`Object` is a valid object identifier but not an existing object:
`existence_error(object,Object)`
`Predicate` is a variable:
`instantiation_error`
`Predicate` is neither a variable nor a predicate indicator:
`type_error(predicate_indicator,Predicate)`
`Predicate` is a valid predicate indicator but not an existing public predicate:
`existence_error(predicate,Predicate)`

---

write_stream/3

Writes a TSV stream with the data represented by the solutions to the specified object predicate.

Compilation flags:
`static`

Template:
`write_stream(Stream,Object,Predicate)`
Mode and number of proofs:
`write_stream(+stream_or_alias,+object_identifier,+predicate_indicator)` - `one`

Exceptions:
`Stream` is a variable:
`instantiation_error`

`Stream` is neither a variable nor a stream-term or alias:
`domain_error(stream_or_alias,Stream)`
`Stream` is not an open stream:
`existence_error(stream,Stream)`
`Stream` is an input stream:
`permission_error(output,stream,Stream)`
`Stream` is a binary stream:
`permission_error(output,binary_stream,Stream)`
`Object` is a variable:
`instantiation_error`
`Object` is neither a variable nor an object identifier:
`type_error(object_identifier,Object)`
`Object` is a valid object identifier but not an existing object:
`existence_error(object,Object)`
`Predicate` is a variable:
`instantiation_error`
`Predicate` is neither a variable nor a predicate indicator:
`type_error(predicate_indicator,Predicate)`
`Predicate` is a valid predicate indicator but not an existing public predicate:
`existence_error(predicate,Predicate)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.337 tutor

object

1.337.1 `tutor`

This object adds explanations and suggestions to selected compiler and developer tool warning and error messages.

Availability:
`logtalk_load(tutor(loader))`

Author: Paulo Moura
Version: 0:86:0
Date: 2026-03-28

Compilation flags:
`static, context_switching_calls`

Imports:
`public` *tutor_explanations*
Provides:
logtalk::message_hook/4
Uses:
list
logtalk

Remarks:

- Usage: Simply load this object at startup using the goal `logtalk_load(tutor(loader))`.

Inherited public predicates:
explain//1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.337.2 `tutor_explanations`

This category provides explanations and suggestions to selected compiler and developer tool warning and error messages.

Availability:
`logtalk_load(tutor(loader))`

Author: Paulo Moura
Version: 0:91:0

Date: 2026-09-30

Compilation flags:
`static`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *explain//1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

explain//1

Generates an explanation for a message.

Compilation flags:
`static`

Template:
`explain(Message)`

Mode and number of proofs:
`explain(@callable)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.338 typeid

object

1.338.1 `typeid`

Type-safe, K-sortable, globally unique identifier (TypeID) generator using an atom representation.

Availability:
`logtalk_load(typeid(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *typeid(atom)*

Remarks:
(none)

Inherited public predicates:
compose/3 decompose/3 from_uuid/3 generate/1 generate/2 generate/3 prefix/2 suffix/2 to_uuid/2 valid/1 valid_prefix/1 valid_suffix/1

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

typeid(Representation), *ids*, *cuid2*, *ksuid*, *nanoid*, *snowflakeid*, *sqids*, *ulid*, *uuid*

object

1.338.2 `typeid(Representation)`

- `Representation` - Text representation for the TypeID, its type prefix, and its UUID suffix. Possible values are `atom`, `chars`, and `codes`.

Type-safe, K-sortable, globally unique identifier (TypeID) generator. Implements the TypeID specification (a type prefix plus a version 7 UUID encoded using a strict, lowercase variant of Crockford base32).

Availability:

`logtalk_load(typeid(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-25

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *typeid_protocol*

Uses:

list
uuid(Representation)

Remarks:

(none)

Inherited public predicates:

compose/3 *decompose/3* *from_uuid/3* *generate/1* *generate/2* *generate/3* *prefix/2* *suffix/2* *to_uuid/2* *valid/1* *valid_prefix/1* *valid_suffix/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

typeid, *ids(Representation,Bytes)*, *cuid2(Representation,Size,Alphabet)*, *ksuid(Representation,Alphabet)*, *nanoid(Representation,Size,Alphabet)*, *snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceB*
sqids(Representation), *ulid(Representation)*, *uuid(Representation)*

protocol

1.338.3 `typeid_protocol`

Type-safe, K-sortable, globally unique identifier (TypeID) generator protocol.

Availability:
`logtalk_load(typeid(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-24

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *generate/1*
  - *generate/2*
  - *generate/3*
  - *from_uuid/3*
  - *to_uuid/2*
  - *prefix/2*
  - *suffix/2*
  - *decompose/3*
  - *compose/3*
  - *valid/1*
  - *valid_prefix/1*
  - *valid_suffix/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

generate/1

Generates a random TypeID with an empty type prefix. The UUID suffix is a version 7 UUID.

Compilation flags:
`static`

Template:
`generate(TypeID)`
Mode and number of proofs:
`generate(--text) - one`

---

generate/2

Generates a random TypeID with the given type prefix. The UUID suffix is a version 7 UUID. Fails if the prefix is not valid per the TypeID specification.

Compilation flags:
`static`

Template:
`generate(Prefix,TypeID)`
Mode and number of proofs:
`generate(+text,--text) - zero_or_one`

generate/3

Generates a random TypeID with the given type prefix. The UUID suffix is a version 7 UUID computed using the given local UTC offset (`Z` or `+HH:MM`/`-HH:MM`) to convert the backend local time to UTC. Fails if the prefix is not valid per the TypeID specification.

Compilation flags:
`static`

Template:
`generate(Prefix,Offset,TypeID)`
Mode and number of proofs:
`generate(+text,+atom,--text)` - `zero_or_one`

---

from_uuid/3

Returns the TypeID for the given type prefix and UUID. The UUID is not required to be a version 7 UUID, allowing encoding of other UUID versions at the user discretion. Fails if the prefix is not valid per the TypeID specification or if the UUID cannot be parsed.

Compilation flags:
`static`

Template:
`from_uuid(Prefix,UUID,TypeID)`
Mode and number of proofs:
`from_uuid(+text,+text,--text)` - `zero_or_one`

---

to_uuid/2

Returns the UUID encoded in the suffix of the given TypeID. Fails if the TypeID is not valid per the TypeID specification.

Compilation flags:
`static`

Template:
`to_uuid(TypeID,UUID)`
Mode and number of proofs:
`to_uuid(+text,--text)` - `zero_or_one`

---

prefix/2

Returns the type prefix of the given TypeID. Fails if the TypeID is not valid per the TypeID specification.

Compilation flags:

```
static
```

Template:

```
prefix(TypeID,Prefix)
```

Mode and number of proofs:

```
prefix(+text,--text) - zero_or_one
```

---

suffix/2

Returns the base32 encoded UUID suffix of the given TypeID. Fails if the TypeID is not valid per the TypeID specification.

Compilation flags:

```
static
```

Template:

```
suffix(TypeID,Suffix)
```

Mode and number of proofs:

```
suffix(+text,--text) - zero_or_one
```

---

decompose/3

Decomposes a TypeID into its type prefix and its base32 encoded UUID suffix. Fails if the TypeID is not valid per the TypeID specification.

Compilation flags:

```
static
```

Template:

```
decompose(TypeID,Prefix,Suffix)
```

Mode and number of proofs:

```
decompose(+text,--text,--text) - zero_or_one
```

---

compose/3

Composes a TypeID from a type prefix and a base32 encoded UUID suffix. Fails if the prefix or the suffix are not valid per the TypeID specification.

Compilation flags:
`static`

Template:
`compose(Prefix,Suffix,TypeID)`
Mode and number of proofs:
`compose(+text,+text,--text)` - `zero_or_one`

---

valid/1

Succeeds if the given TypeID is valid per the TypeID specification.

Compilation flags:
`static`

Template:
`valid(TypeID)`
Mode and number of proofs:
`valid(+text)` - `zero_or_one`

---

valid_prefix/1

Succeeds if the given type prefix is valid per the TypeID specification.

Compilation flags:
`static`

Template:
`valid_prefix(Prefix)`
Mode and number of proofs:
`valid_prefix(+text)` - `zero_or_one`

---

valid_suffix/1

Succeeds if the given base32 encoded UUID suffix is valid per the TypeID specification.

Compilation flags:
```
static
```

Template:
```
valid_suffix(Suffix)
```
Mode and number of proofs:
```
valid_suffix(+text) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.339 types

object

1.339.1 atom

Atom data type predicates.

Availability:
```
logtalk_load(types(loader))
```

Author: Paulo Moura
Version: 1:9:0
Date: 2023-04-12

Compilation flags:
```
static, context_switching_calls
```

Extends:
```
public atomic
```
Uses:
```
user
```

Remarks:

(none)

Inherited public predicates:

(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 check/1 depth/2 ground/1 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 subsumes/2 subterm/2 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
  - *replace_sub_atom/4*
  - *split/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

replace_sub_atom/4

Replaces all occurrences of `Old` by `New` in `Input` returning `Output`. Returns `Input` if `Old` is the empty atom. Fails when `Output` does not unify with the resulting atom.

Compilation flags:

```
static
```

Template:

```
replace_sub_atom(Old,New,Input,Output)
```

Mode and number of proofs:

```
replace_sub_atom(+atom,+atom,+atom,?atom) - zero_or_one
```

---

split/3

Splits an atom at a given delimiter into a list of sub-atoms.

Compilation flags:

```
static
```

Template:

```
split(Atom,Delimiter,SubAtoms)
```

Mode and number of proofs:

```
split(+atom,+atom,-list(atom)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.339.2 `atomic`

Atomic data type predicates.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2018-07-11

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *term*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 check/1 depth/2 ground/1 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 subsumes/2 subterm/2 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.339.3 `callable`

Callable term type predicates.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:4:0
Date: 2018-07-11

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *term*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 check/1 depth/2 ground/1 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 subsumes/2 subterm/2 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.339.4 `character`

Character predicates (most of them assume an ASCII representation).

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:9:0
Date: 2019-06-29

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *characterp*
Extends:
`public` *atom*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 check/1 depth/2 ground/1 is_alpha/1 is_alphanumeric/1 is_ascii/1 is_bin_digit/1 is_control/1 is_dec_digit/1 is_end_of_line/1 is_hex_digit/1 is_layout/1 is_letter/1 is_lower_case/1 is_newline/1 is_octal_digit/1 is_period/1 is_punctuation/1 is_quote/1 is_upper_case/1 is_vowel/1 is_white_space/1 lower_upper/2 new/1 numbervars/1 numbervars/3 occurs/2 parenthesis/2 replace_sub_atom/4 singletons/2 split/3 subsumes/2 subterm/2 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.339.5 `characterp`

Character protocol.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2019-06-29

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *is_ascii/1*
  - *is_alphanumeric/1*
  - *is_alpha/1*
  - *is_letter/1*
  - *is_bin_digit/1*
  - *is_octal_digit/1*
  - *is_dec_digit/1*
  - *is_hex_digit/1*

      - *is_lower_case/1*
      - *is_upper_case/1*
      - *is_vowel/1*
      - *is_white_space/1*
      - *is_layout/1*
      - *is_quote/1*
      - *is_punctuation/1*
      - *is_period/1*
      - *is_control/1*
      - *is_newline/1*
      - *is_end_of_line/1*
      - *parenthesis/2*
      - *lower_upper/2*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

is_ascii/1

True if the argument is an ASCII character.

Compilation flags:
`static`

Template:
`is_ascii(Char)`
Mode and number of proofs:
`is_ascii(+char)` - `zero_or_one`

---

is_alphanumeric/1

True if the argument is an alphanumeric character.

Compilation flags:
`static`

Template:
`is_alphanumeric(Char)`
Mode and number of proofs:
`is_alphanumeric(+char)` - `zero_or_one`

---

is_alpha/1

True if the argument is a letter or an underscore.

Compilation flags:
`static`

Template:
`is_alpha(Char)`
Mode and number of proofs:
`is_alpha(+char)` - `zero_or_one`

---

is_letter/1

True if the argument is a letter.

Compilation flags:
`static`

Template:
`is_letter(Char)`
Mode and number of proofs:
`is_letter(+char)` - `zero_or_one`

---

is_bin_digit/1

True if the argument is a binary digit.

Compilation flags:
`static`

Template:
`is_bin_digit(Char)`
Mode and number of proofs:
`is_bin_digit(+char)` - `zero_or_one`

---

is_octal_digit/1

True if the argument is an octal digit.

Compilation flags:
`static`

Template:
`is_octal_digit(Char)`
Mode and number of proofs:
`is_octal_digit(+char)` - `zero_or_one`

---

is_dec_digit/1

True if the argument is a decimal digit.

Compilation flags:
`static`

Template:
`is_dec_digit(Char)`
Mode and number of proofs:
`is_dec_digit(+char)` - `zero_or_one`

---

is_hex_digit/1

True if the argument is an hexadecimal digit.

Compilation flags:
`static`

Template:
`is_hex_digit(Char)`
Mode and number of proofs:
`is_hex_digit(+char)` - `zero_or_one`

---

is_lower_case/1

True if the argument is a lower case letter.

Compilation flags:
`static`

Template:
`is_lower_case(Char)`
Mode and number of proofs:
`is_lower_case(+char)` - `zero_or_one`

---

is_upper_case/1

True if the argument is a upper case letter.

Compilation flags:
`static`

Template:
`is_upper_case(Char)`
Mode and number of proofs:
`is_upper_case(+char)` - `zero_or_one`

---

is_vowel/1

True if the argument is a vowel.

Compilation flags:
`static`

Template:
`is_vowel(Char)`
Mode and number of proofs:
`is_vowel(+char)` - `zero_or_one`

---

is_white_space/1

True if the argument is a white space character (a space or a tab) inside a line of characters.

Compilation flags:
`static`

Template:
`is_white_space(Char)`
Mode and number of proofs:
`is_white_space(+char)` - `zero_or_one`

---

is_layout/1

True if the argument is a layout character.

Compilation flags:
`static`

Template:
`is_layout(Char)`
Mode and number of proofs:
`is_layout(+char)` - `zero_or_one`

---

is_quote/1

True if the argument is a quote character.

Compilation flags:
`static`

Template:
`is_quote(Char)`
Mode and number of proofs:
`is_quote(+char)` - `zero_or_one`

---

is_punctuation/1

True if the argument is a sentence punctuation character.

Compilation flags:
`static`

Template:
`is_punctuation(Char)`
Mode and number of proofs:
`is_punctuation(+char)` - `zero_or_one`

---

is_period/1

True if the argument is a character that ends a sentence.

Compilation flags:
`static`

Template:
`is_period(Char)`
Mode and number of proofs:
`is_period(+char)` - `zero_or_one`

---

is_control/1

True if the argument is an ASCII control character.

Compilation flags:
`static`

Template:
`is_control(Char)`
Mode and number of proofs:
`is_control(+char)` - `zero_or_one`

---

is_newline/1

True if the argument is the ASCII newline character.

Compilation flags:
`static`

Template:
`is_newline(Char)`
Mode and number of proofs:
`is_newline(+char)` - `zero_or_one`

---

is_end_of_line/1

True if the argument is the ASCII end-of-line character (either a carriage return or a line feed).

Compilation flags:
`static`

Template:
`is_end_of_line(Char)`
Mode and number of proofs:
`is_end_of_line(+char)` - `zero_or_one`

---

parenthesis/2

Recognizes and converts between open and close parenthesis.

Compilation flags:
`static`

Template:
`parenthesis(Char1,Char2)`
Mode and number of proofs:
`parenthesis(?char,?char)` - `zero_or_more`
`parenthesis(+char,?char)` - `zero_or_one`
`parenthesis(?char,+char)` - `zero_or_one`

`lower_upper/2`

Recognizes and converts between lower and upper case letters.

Compilation flags:
`static`

Template:
`lower_upper(Char1,Char2)`
Mode and number of proofs:
`lower_upper(?char,?char)` - `zero_or_more`
`lower_upper(+char,?char)` - `zero_or_one`
`lower_upper(?char,+char)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

character

protocol

1.339.6 `comparingp`

Comparing protocol using overloading of standard operators.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2000-07-24

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *(<)/2*
  - *(=<)/2*
  - *(>)/2*
  - *(>=)/2*
  - *(=:=)/2*
  - *(=\=)/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(<)/2

True if `Term1` is less than `Term2`.

Compilation flags:
`static`

Template:
`Term1<Term2`
Mode and number of proofs:
`+term< +term - zero_or_one`

(=<)/2

True if `Term1` is less or equal than `Term2`.

Compilation flags:
`static`

Template:
`Term1=<Term2`
Mode and number of proofs:
`+term=< +term - zero_or_one`

(>)/2

True if Term1 is greater than Term2.

Compilation flags:
static

Template:
Term1>Term2
Mode and number of proofs:
+term> +term - zero_or_one

(>=)/2

True if Term1 is equal or grater than Term2.

Compilation flags:
static

Template:
Term1>=Term2
Mode and number of proofs:
+term>= +term - zero_or_one

(=:=)/2

True if Term1 is equal to Term2.

Compilation flags:
static

Template:
Term1=:=Term2
Mode and number of proofs:
+term=:= +term - zero_or_one

`(=\=)/2`

True if `Term1` is not equal to `Term2`.

Compilation flags:
`static`

Template:
`Term1=\=Term2`
Mode and number of proofs:
`+term=\= +term - zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.339.7 `compound`

Compound data type.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2018-07-11

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *term*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 check/1 depth/2 ground/1 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 subsumes/2 subterm/2 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.339.8 `difflist`

Difference list predicates.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 2:1:0
Date: 2026-02-03

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *listp*
Extends:
`public` *compound*
Uses:
list

Remarks:
(none)

Inherited public predicates:

(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 append/2 append/3 check/1 delete/3 delete_matches/3 depth/2 drop/3 empty/1 flatten/2 ground/1 hamming_distance/3 keysort/2 last/2 length/2 max/2 member/2 memberchk/2 min/2 msort/2 msort/3 new/1 nextto/3 nth0/3 nth0/4 nth1/3 nth1/4 numbervars/1 numbervars/3 occurrences/2 occurrences/3 occurs/2 partition/5 permutation/2 prefix/2 prefix/3 proper_prefix/2 proper_prefix/3 proper_suffix/2 proper_suffix/3 remove_duplicates/2 reverse/2 same_length/2 same_length/3 select/3 select/4 selectchk/3 selectchk/4 sequential_occurrences/2 sequential_occurrences/3 singletons/2 sort/2 sort/3 sort/4 split/4 sublist/2 subsequence/3 subsequence/4 substitute/4 subsumes/2 subterm/2 subtract/3 suffix/2 suffix/3 take/3 take/4 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
  - *add/3*
  - *as_list/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

add/3

Adds a term to the end of a difference list.

Compilation flags:
```
static
```

Template:
```
add(Term,DiffList,NewDiffList)
```
Mode and number of proofs:
```
add(@term,+difference_list,-difference_list) - one
```

---

as_list/2

Returns a list with the elements of the difference list.

Compilation flags:
```
static
```

Template:
```
as_list(DiffList,List)
```
Mode and number of proofs:
```
as_list(@difference_list,-list) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

list, *list(Type)*, *numberlist*, *varlist*

object

1.339.9 `float`

Floating point numbers data type predicates.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:7:0
Date: 2025-02-25

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *number*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 =~= /2 (>)/2 (>=)/2 approximately_equal/2 approximately_equal/3 check/1 depth/2 essentially_equal/3 ground/1 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 subsumes/2 subterm/2 tolerance_equal/4 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
  - `between/4`
  - `sequence/4`
  - `sequence/5`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

between/4

Enumerates by backtracking a sequence of `N` equally spaced floats in the interval `[Lower,Upper]`. Assumes `N > 0` and `Lower =< Upper`; fails otherwise.

Compilation flags:

```
static
```

Template:

```
between(Lower,Upper,N,Float)
```

Mode and number of proofs:

```
between(+float,+float,+positive_integer,-float) - zero_or_more
```

---

sequence/4

Generates a list with the sequence of `N` equally spaced floats in the interval `[Lower,Upper]`. Assumes `N > 0` and `Lower =< Upper`; fails otherwise.

Compilation flags:

```
static
```

Template:

```
sequence(Lower,Upper,N,List)
```

Mode and number of proofs:

```
sequence(+float,+float,+positive_integer,-list(float)) - zero_or_one
```

---

sequence/5

Generates a list with the sequence of `Step` spaced floats in the interval `[Lower,Upper]`. Also returns the length of the list. Assumes `Lower =< Upper`; fails otherwise.

Compilation flags:

```
static
```

Template:

```
sequence(Lower,Upper,Step,List,Length)
```

Mode and number of proofs:

```
sequence(+float,+float,+float,-list(float),-positive_integer) - zero_or_one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.339.10 `integer`

Integer data type predicates.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:56:0
Date: 2025-05-26

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *number*

Remarks:

- Portability notes: This object will use the backend Prolog system `between/3`, `plus/3`, and `succ/2` built-in predicates when available.

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 =~= /2 (>)/2 (>=)/2 approximately_equal/2 approximately_equal/3 check/1 depth/2 essentially_equal/3 ground/1 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 subsumes/2 subterm/2 tolerance_equal/4 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
  - *between/3*
  - *plus/3*
  - *succ/2*
  - *sequence/3*
  - *sequence/4*
  - *power_sequence/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

between/3

Returns integers in the interval defined by the two first arguments.

Compilation flags:

```
static
```

Template:

```
between(Lower,Upper,Integer)
```

Mode and number of proofs:

```
between(+integer,+integer,+integer) - zero_or_one
between(+integer,+integer,-integer) - zero_or_more
```

---

plus/3

Reversible integer sum. At least two of the arguments must be instantiated to integers.

Compilation flags:

```
static
```

Template:

```
plus(I,J,Sum)
```

Mode and number of proofs:

```
plus(+integer,+integer,?integer) - zero_or_one
plus(+integer,?integer,+integer) - zero_or_one
plus(?integer,+integer,+integer) - zero_or_one
```

---

succ/2

Successor of a natural number. At least one of the arguments must be instantiated to a natural number.

Compilation flags:

```
static
```

Template:

```
succ(I,J)
```

Mode and number of proofs:

```
succ(+integer,?integer) - zero_or_one
succ(?integer,+integer) - zero_or_one
```

---

sequence/3

Generates a list with the sequence of all integers in the interval `[Lower,Upper]`. Assumes `Lower =< Upper` and fails otherwise.

Compilation flags:
`static`

Template:
`sequence(Lower,Upper,List)`
Mode and number of proofs:
`sequence(+integer,+integer,-list(integer))` - `zero_or_one`

---

sequence/4

Generates a list with the sequence of integers in the interval `[Lower,Upper]` by `Step`. Assumes `Lower =< Upper, Step >= 1` and fails otherwise.

Compilation flags:
`static`

Template:
`sequence(Lower,Upper,Step,List)`
Mode and number of proofs:
`sequence(+integer,+integer,+integer,-list(integer))` - `zero_or_one`

---

power_sequence/4

Generates a list of exponentiation results given `[Lower,Upper]` sequence of exponents and `Base`. Assumes `Lower =< Upper` and `Base > 1` and fails otherwise.

Compilation flags:
`static`

Template:
`power_sequence(Lower,Upper,Base,List)`
Mode and number of proofs:
`power_sequence(+integer,+integer,+integer,-list(integer))` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.339.11 `list`

List predicates.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 4:5:0
Date: 2026-08-05

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *listp*
Extends:
`public` *compound*

Remarks:

- Portability notes: This object will use the backend Prolog system `msort/2` and `sort/4` built-in predicates when available.

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 append/2 append/3 check/1 delete/3 delete_matches/3 depth/2 drop/3 empty/1 flatten/2 ground/1 hamming_distance/3 keysort/2 last/2 length/2 max/2 member/2 memberchk/2 min/2 msort/2 msort/3 new/1 nextto/3 nth0/3 nth0/4 nth1/3 nth1/4 numbervars/1 numbervars/3 occurrences/2 occurrences/3 occurs/2 partition/5 permutation/2 prefix/2 prefix/3 proper_prefix/2 proper_prefix/3 proper_suffix/2 proper_suffix/3 remove_duplicates/2 reverse/2 same_length/2 same_length/3 select/3 select/4 selectchk/3 selectchk/4 sequential_occurrences/2 sequential_occurrences/3 singletons/2 sort/2 sort/3 sort/4 split/4 sublist/2 subsequence/3 subsequence/4 substitute/4 subsumes/2 subterm/2 subtract/3 suffix/2 suffix/3 take/3 take/4 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
  - *as_difflist/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`as_difflist/2`

Converts a list to a difference list.

Compilation flags:
`static`

Template:
`as_difflist(List,Diffist)`
Mode and number of proofs:
`as_difflist(+list,-difference_list) - one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

list(Type), *numberlist*, *varlist*, *difflist*

object

1.339.12 `list(Type)`

List predicates with elements constrained to a single type.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:22:0
Date: 2018-07-11

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *list*

Remarks:

(none)

Inherited public predicates:

(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 append/2 append/3 as_difflist/2 check/1 delete/3 delete_matches/3 depth/2 drop/3 empty/1 flatten/2 ground/1 hamming_distance/3 keysort/2 last/2 length/2 max/2 member/2 memberchk/2 min/2 msort/2 msort/3 new/1 nextto/3 nth0/3 nth0/4 nth1/3 nth1/4 numbervars/1 numbervars/3 occurrences/2 occurrences/3 occurs/2 partition/5 permutation/2 prefix/2 prefix/3 proper_prefix/2 proper_prefix/3 proper_suffix/2 proper_suffix/3 remove_duplicates/2 reverse/2 same_length/2 same_length/3 select/3 select/4 selectchk/3 selectchk/4 sequential_occurrences/2 sequential_occurrences/3 singletons/2 sort/2 sort/3 sort/4 split/4 sublist/2 subsequence/3 subsequence/4 substitute/4 subsumes/2 subterm/2 subtract/3 suffix/2 suffix/3 take/3 take/4 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

list, *numberlist*, *varlist*, *difflist*

protocol

1.339.13 `listp`

List protocol.

Availability:

```
logtalk_load(types(loader))
```

Author: Paulo Moura
Version: 1:20:0
Date: 2026-09-04

Compilation flags:

`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *append/2*
  - *append/3*
  - *delete/3*
  - *delete_matches/3*
  - *empty/1*
  - *flatten/2*
  - *hamming_distance/3*
  - *keysort/2*
  - *last/2*
  - *length/2*
  - *max/2*
  - *member/2*
  - *memberchk/2*
  - *min/2*
  - *msort/2*
  - *msort/3*
  - *nextto/3*
  - *nth0/3*
  - *nth0/4*
  - *nth1/3*
  - *nth1/4*
  - *sequential_occurrences/2*
  - *sequential_occurrences/3*
  - *occurrences/2*
  - *occurrences/3*
  - *partition/5*
  - *permutation/2*
  - *prefix/2*
  - *prefix/3*
  - *proper_prefix/2*
  - *proper_prefix/3*
  - *remove_duplicates/2*
  - *reverse/2*
  - *same_length/2*
  - *same_length/3*

      - *select/3*
      - *selectchk/3*
      - *select/4*
      - *selectchk/4*
      - *sort/2*
      - *sort/3*
      - *sort/4*
      - *split/4*
      - *sublist/2*
      - *subsequence/3*
      - *subsequence/4*
      - *substitute/4*
      - *subtract/3*
      - *suffix/2*
      - *suffix/3*
      - *proper_suffix/2*
      - *proper_suffix/3*
      - *take/3*
      - *take/4*
      - *drop/3*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

append/2

Appends all lists in a list of lists.

Compilation flags:
    `static`

Template:
    `append(Lists,Concatenation)`
Mode and number of proofs:
    `append(+list(list),?list)` - `zero_or_one`

---

append/3

Appends two lists.

Compilation flags:
`static`

Template:
`append(List1,List2,List)`
Mode and number of proofs:
`append(?list,?list,?list)` - `zero_or_more`

---

delete/3

Deletes from a list all occurrences of an element returning the list of remaining elements. Uses `==/2` for element comparison.

Compilation flags:
`static`

Template:
`delete(List,Element,Remaining)`
Mode and number of proofs:
`delete(@list,@term,?list)` - `one`

---

delete_matches/3

Deletes all matching elements from a list, returning the list of remaining elements. Uses `=/2` for element comparison.

Compilation flags:
`static`

Template:
`delete_matches(List,Element,Remaining)`
Mode and number of proofs:
`delete_matches(@list,@term,?list)` - `one`

---

empty/1

True if the argument is an empty list.

Compilation flags:
`static`

Template:
`empty(List)`
Mode and number of proofs:
`empty(@list)` - `zero_or_one`

---

flatten/2

Flattens a list of lists into a list.

Compilation flags:
`static`

Template:
`flatten(List,Flatted)`
Mode and number of proofs:
`flatten(+list,-list)` - `one`

---

hamming_distance/3

Calculates the Hamming distance between two lists (using equality to compare list elements). Fails if the two lists are not of the same length.

Compilation flags:
`static`

Template:
`hamming_distance(List1,List2,Distance)`
Mode and number of proofs:
`hamming_distance(+list,+list,-integer)` - `zero_or_one`

---

keysort/2

Sorts a list of key-value pairs in ascending order.

Compilation flags:
`static`

Template:
`keysort(List,Sorted)`
Mode and number of proofs:
`keysort(+list(pair),-list(pair))` - `one`

---

last/2

List last element (if it exists).

Compilation flags:
`static`

Template:
`last(List,Last)`
Mode and number of proofs:
`last(?list,?term)` - `zero_or_more`

---

length/2

List length.

Compilation flags:
`static`

Template:
`length(List,Length)`
Mode and number of proofs:
`length(?list,?integer)` - `zero_or_more`

---

max/2

Determines the list maximum value using standard order. Fails if the list is empty.

Compilation flags:
```
static
```

Template:
```
max(List,Maximum)
```
Mode and number of proofs:
```
max(+list,-term) - zero_or_one
```

---

member/2

Element is a list member.

Compilation flags:
```
static
```

Template:
```
member(Element,List)
```
Mode and number of proofs:
```
member(?term,?list) - zero_or_more
```

---

memberchk/2

Checks if a term is a member of a list.

Compilation flags:
```
static
```

Template:
```
memberchk(Element,List)
```
Mode and number of proofs:
```
memberchk(?term,?list) - zero_or_one
```

---

min/2

Determines the minimum value in a list using standard order. Fails if the list is empty.

Compilation flags:
```
static
```

Template:
```
min(List,Minimum)
```
Mode and number of proofs:
```
min(+list,-term) - zero_or_one
```

---

msort/2

Sorts a list in ascending order (duplicated elements are not removed).

Compilation flags:
```
static
```

Template:
```
msort(List,Sorted)
```
Mode and number of proofs:
```
msort(+list,-list) - one
```

---

msort/3

Sorts a list using a user-specified comparison predicate modeled on the standard `compare/3` predicate (duplicated elements are not removed).

Compilation flags:
```
static
```

Template:
```
msort(Closure,List,Sorted)
```
Meta-predicate template:
```
msort(3,*,*)
```
Mode and number of proofs:
```
msort(+callable,+list,-list) - one
```

---

nextto/3

`X` and `Y` are consecutive elements in List.

Compilation flags:
`static`

Template:
`nextto(X,Y,List)`
Mode and number of proofs:
`nextto(?term,?term,?list)` - `zero_or_more`

---

nth0/3

Nth element of a list (counting from zero).

Compilation flags:
`static`

Template:
`nth0(Nth,List,Element)`
Mode and number of proofs:
`nth0(?integer,?list,?term)` - `zero_or_more`

---

nth0/4

Nth element of a list (counting from zero). `Rest` is a list of all the other elements. Can be used to either select the nth element of `List` or to insert an element before the nth element in `Rest`.

Compilation flags:
`static`

Template:
`nth0(Nth,List,Element,Rest)`
Mode and number of proofs:
`nth0(?integer,?list,?term,?list)` - `zero_or_more`

---

nth1/3

Nth element of a list (counting from one).

Compilation flags:
`static`

Template:
`nth1(Nth,List,Element)`
Mode and number of proofs:
`nth1(?integer,?list,?term)` - `zero_or_more`

---

nth1/4

Nth element of a list (counting from one). `Rest` is a list of all the other elements. Can be used to either select the nth element of `List` or to insert an element before the nth element in `Rest`.

Compilation flags:
`static`

Template:
`nth1(Nth,List,Element,Rest)`
Mode and number of proofs:
`nth1(?integer,?list,?term,?list)` - `zero_or_more`

---

sequential_occurrences/2

Counts the number of sequential occurrences of each `List` element, unifying `Occurrences` with a list of `Element-Count` pairs. Uses term equality for element comparison.

Compilation flags:
`static`

Template:
`sequential_occurrences(List,Occurrences)`
Mode and number of proofs:
`sequential_occurrences(@list,-list(pair(term,positive_integer)))` - `one`

---

sequential_occurrences/3

Counts the number of sequential occurrences of each `List` element, unifying `Occurrences` with a list of `Element-Count` pairs. Uses `Closure` for element comparison.

Compilation flags:
`static`

Template:
`sequential_occurrences(List,Closure,Occurrences)`
Mode and number of proofs:
`sequential_occurrences(@list,@callable,-list(pair(term,positive_integer)))` - `one`

---

occurrences/2

Counts the number of occurrences of each `List` element, unifying `Occurrences` with a sorted list of `Element-Count` pairs. Uses term equality for element comparison.

Compilation flags:
`static`

Template:
`occurrences(List,Occurrences)`
Mode and number of proofs:
`occurrences(@list,-list(pair(term,positive_integer)))` - `one`

---

occurrences/3

Counts the number of occurrences of each `List` element, unifying `Occurrences` with a sorted list of `Element-Count` pairs. Uses `Closure` for element comparison.

Compilation flags:
`static`

Template:
`occurrences(List,Closure,Occurrences)`
Meta-predicate template:
`occurrences(*,2,*)`
Mode and number of proofs:
`occurrences(@list,@callable,-list(pair(term,positive_integer)))` - `one`

---

partition/5

Partitions a list in lists with values less, equal, and greater than a given value (using standard order).

Compilation flags:
`static`

Template:
`partition(List,Value,Less,Equal,Greater)`
Mode and number of proofs:
`partition(+list,+number,-list,-list,-list)` - `one`

---

permutation/2

The two lists are a permutation of the same list.

Compilation flags:
`static`

Template:
`permutation(List,Permutation)`
Mode and number of proofs:
`permutation(?list,?list)` - `zero_or_more`

---

prefix/2

`Prefix` is a prefix of `List`.

Compilation flags:
`static`

Template:
`prefix(Prefix,List)`
Mode and number of proofs:
`prefix(?list,+list)` - `zero_or_more`

---

prefix/3

`Prefix` is a prefix of length `Length` of `List`.

Compilation flags:
`static`

Template:
`prefix(Prefix,Length,List)`
Mode and number of proofs:
`prefix(?list,+integer,+list)` - `zero_or_one`
`prefix(?list,-integer,+list)` - `zero_or_more`

---

proper_prefix/2

`Prefix` is a proper prefix of `List`.

Compilation flags:
`static`

Template:
`proper_prefix(Prefix,List)`
Mode and number of proofs:
`proper_prefix(?list,+list)` - `zero_or_more`

---

proper_prefix/3

`Prefix` is a proper prefix of length `Length` of `List`.

Compilation flags:
`static`

Template:
`proper_prefix(Prefix,Length,List)`
Mode and number of proofs:
`proper_prefix(?list,+integer,+list)` - `zero_or_one`
`proper_prefix(?list,-integer,+list)` - `zero_or_more`

---

remove_duplicates/2

Removes duplicated list elements using equality (==/2) for comparison and keeping the left-most element when repeated.

Compilation flags:
```
static
```

Template:
```
remove_duplicates(List,Set)
```
Mode and number of proofs:
```
remove_duplicates(+list,-list) - one
```

---

reverse/2

Reverses a list.

Compilation flags:
```
static
```

Template:
```
reverse(List,Reversed)
```
Mode and number of proofs:
```
reverse(+list,?list) - zero_or_one
reverse(?list,+list) - zero_or_one
reverse(-list,-list) - one_or_more
```

---

same_length/2

The two lists have the same length.

Compilation flags:
```
static
```

Template:
```
same_length(List1,List2)
```
Mode and number of proofs:
```
same_length(+list,?list) - zero_or_one
same_length(?list,+list) - zero_or_one
same_length(-list,-list) - one_or_more
```

---

same_length/3

The two lists have the same length.

Compilation flags:
```
static
```

Template:
```
same_length(List1,List2,Length)
```
Mode and number of proofs:
```
same_length(+list,?list,?integer) - zero_or_one
same_length(?list,+list,?integer) - zero_or_one
same_length(-list,-list,-integer) - one_or_more
```

---

select/3

Selects an element from a list, returning the list of remaining elements.

Compilation flags:
```
static
```

Template:
```
select(Element,List,Remaining)
```
Mode and number of proofs:
```
select(?term,?list,?list) - zero_or_more
```

---

selectchk/3

Checks that an element can be selected from a list, returning the list of remaining elements.

Compilation flags:
```
static
```

Template:
```
selectchk(Element,List,Remaining)
```
Mode and number of proofs:
```
selectchk(?term,?list,?list) - zero_or_one
```

---

select/4

Selects an element from a list, replacing it by a new element and returning the resulting list.

Compilation flags:
`static`

Template:
`select(Old,OldList,New,NewList)`
Mode and number of proofs:
`select(?term,?list,?term,?list)` - `zero_or_more`

---

selectchk/4

Checks that an element from a list can be replaced by a new element, returning the resulting list.

Compilation flags:
`static`

Template:
`selectchk(Old,OldList,New,NewList)`
Mode and number of proofs:
`selectchk(?term,?list,?term,?list)` - `zero_or_one`

---

sort/2

Sorts a list in ascending order (duplicated elements are removed).

Compilation flags:
`static`

Template:
`sort(List,Sorted)`
Mode and number of proofs:
`sort(+list,-list)` - `one`

---

sort/3

Sorts a list using a user-specified comparison predicate modeled on the standard `compare/3` predicate (duplicated elements are removed).

Compilation flags:
`static`

Template:
`sort(Closure,List,Sorted)`
Meta-predicate template:
`sort(3,*,*)`
Mode and number of proofs:
`sort(+callable,+list,-list)` - `one`

---

sort/4

Sorts a list using the given key and order. Uses the standard term comparison operators for the order. The key selects the argument in each element in the list to use for comparisons. A key value of zero uses the whole element for comparisons.

Compilation flags:
`static`

Template:
`sort(Key,Order,List,Sorted)`
Mode and number of proofs:
`sort(+non_negative_integer,+atom,+list,-list)` - `one`

Remarks:

- Removing duplicates: Use one of the `@<` or `@>` orders.
- Keeping duplicates: Use one of the `@=<` or `@>=` orders.
- Sorting in ascending order: Use one of the `@<` or `@=<` orders.
- Sorting in descending order: Use one of the `@>` or `@>=` orders.

---

split/4

Splits a list into sublists of a given length. Also returns a list with the remaining elements. Fails if the length is zero or negative.

Compilation flags:
`static`

Template:
`split(List,Length,Sublists,Remaining)`
Mode and number of proofs:
`split(+list,+integer,-list(list),-list)` - `zero_or_one`

---

sublist/2

The first list is a sublist of the second.

Compilation flags:
`static`

Template:
`sublist(Sublist,List)`
Mode and number of proofs:
`sublist(?list,+list)` - `zero_or_more`

---

subsequence/3

`List` is an interleaving of `Subsequence` and `Remaining`. Element order is preserved.

Compilation flags:
`static`

Template:
`subsequence(List,Subsequence,Remaining)`
Mode and number of proofs:
`subsequence(?list,?list,?list)` - `zero_or_more`

---

subsequence/4

Generates subsequences of a given length from a list. Also returns the remaining elements. Element order is preserved.

Compilation flags:
`static`

Template:
`subsequence(List,Length,Subsequence,Remaining)`
Mode and number of proofs:
`subsequence(+list,+integer,?list,?list)` - `zero_or_more`

---

substitute/4

Substitutes all occurrences of `Old` in `List` by `New`, returning `NewList`. Uses term equality for element comparison.

Compilation flags:
`static`

Template:
`substitute(Old,List,New,NewList)`
Mode and number of proofs:
`substitute(@term,@list,@term,-list)` - `one`

---

subtract/3

Removes all elements in the second list from the first list, returning the list of remaining elements.

Compilation flags:
`static`

Template:
`subtract(List,Elements,Remaining)`
Mode and number of proofs:
`subtract(+list,+list,-list)` - `one`

---

suffix/2

Suffix is a suffix of List.

Compilation flags:
`static`

Template:
`suffix(Suffix,List)`
Mode and number of proofs:
`suffix(?list,+list)` - `zero_or_more`

---

suffix/3

`Suffix` is a suffix of length `Length` of `List`.

Compilation flags:
`static`

Template:
`suffix(Suffix,Length,List)`
Mode and number of proofs:
`suffix(?list,+integer,+list)` - `zero_or_one`
`suffix(?list,-integer,+list)` - `zero_or_more`

---

proper_suffix/2

`Suffix` is a proper suffix of `List`.

Compilation flags:
`static`

Template:
`proper_suffix(Suffix,List)`
Mode and number of proofs:
`proper_suffix(?list,+list)` - `zero_or_more`

---

proper_suffix/3

`Suffix` is a proper suffix of length `Length` of `List`.

Compilation flags:
`static`

Template:
`proper_suffix(Suffix,Length,List)`
Mode and number of proofs:
`proper_suffix(?list,+integer,+list)` - `zero_or_one`
`proper_suffix(?list,-integer,+list)` - `zero_or_more`

---

take/3

Takes the first N elements of a list. Fails if the list have fewer than N elements.

Compilation flags:
`static`

Template:
`take(N,List,Elements)`
Mode and number of proofs:
`take(+integer,+list,-list)` - `zero_or_one`

---

take/4

Takes the first N elements of a list (i.e., splits a list after its first N elements). Fails if the list have fewer than N elements.

Compilation flags:
`static`

Template:
`take(N,List,Prefix,Suffix)`
Mode and number of proofs:
`take(+integer,+list,-list,-list)` - `zero_or_one`

---

drop/3

Drops the first N elements of a list. Fails if the list have fewer than N elements.

Compilation flags:
`static`

Template:
`drop(N,List,Remaining)`
Mode and number of proofs:
`drop(+integer,+list,-list)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

list, *list(Type)*, *numberlistp*, *varlistp*

object

1.339.14 `natural`

Natural numbers data type predicates.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 2:3:0
Date: 2026-05-13

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *integer*
Uses:
list
numberlist

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 =~= / 2 (>)/2 (>=)/2 approximately_equal/2 approximately_equal/3 between/3 check/1 depth/2 essentially_equal/3 ground/1 new/1 numbervars/1 numbervars/3 occurs/2 plus/3 power_sequence/4 sequence/3 sequence/4 singletons/2 subsumes/2 subterm/2 succ/2 tolerance_equal/4 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
  - *factorial/2*
  - *fibonacci/2*

      - *lucas/2*
      - *falling_factorial/3*
      - *rising_factorial/3*
      - *catalan/2*
      - *subfactorial/2*
      - *bell/2*
      - *stirling_first/3*
      - *stirling_second/3*
      - *partition_number/2*
      - *binomial/3*
      - *multinomial/2*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

factorial/2

Computes the factorial of a non-negative integer.

Compilation flags:
`static`

Template:
`factorial(N,Factorial)`

Mode and number of proofs:
`factorial(+non_negative_integer,-non_negative_integer)` - `one`

---

fibonacci/2

Computes the Fibonacci number of a non-negative integer.

Compilation flags:
`static`

Template:
`fibonacci(N,Fibonacci)`

Mode and number of proofs:
`fibonacci(+non_negative_integer,-non_negative_integer)` - `one`

---

lucas/2

Computes the Lucas number of a non-negative integer.

Compilation flags:
`static`

Template:
`lucas(N,Lucas)`
Mode and number of proofs:
`lucas(+non_negative_integer,-non_negative_integer)` - `one`

---

falling_factorial/3

Computes the falling factorial. `N` must be greater than or equal to `K` (fails otherwise).

Compilation flags:
`static`

Template:
`falling_factorial(N,K,FallingFactorial)`
Mode and number of proofs:
`falling_factorial(+non_negative_integer,+non_negative_integer,`
`-non_negative_integer)` - `zero_or_one`

---

rising_factorial/3

Computes the rising factorial.

Compilation flags:
`static`

Template:
`rising_factorial(N,K,RisingFactorial)`
Mode and number of proofs:
`rising_factorial(+non_negative_integer,+non_negative_integer,`
`-non_negative_integer)` - `one`

---

catalan/2

Computes the Catalan number of a non-negative integer.

Compilation flags:
```
static
```

Template:
```
catalan(N,Catalan)
```
Mode and number of proofs:
```
catalan(+non_negative_integer,-non_negative_integer) - one
```

---

subfactorial/2

Computes the subfactorial of a non-negative integer.

Compilation flags:
```
static
```

Template:
```
subfactorial(N,Subfactorial)
```
Mode and number of proofs:
```
subfactorial(+non_negative_integer,-non_negative_integer) - one
```

---

bell/2

Computes the Bell number of a non-negative integer.

Compilation flags:
```
static
```

Template:
```
bell(N,Bell)
```
Mode and number of proofs:
```
bell(+non_negative_integer,-non_negative_integer) - one
```

---

stirling_first/3

Computes the unsigned Stirling number of the first kind. N must be greater than or equal to K (fails otherwise).

Compilation flags:
static

Template:
stirling_first(N,K,Stirling)
Mode and number of proofs:
stirling_first(+non_negative_integer,+non_negative_integer,-non_negative_integer) - zero_or_one

---

stirling_second/3

Computes the Stirling number of the second kind. N must be greater than or equal to K (fails otherwise).

Compilation flags:
static

Template:
stirling_second(N,K,Stirling)
Mode and number of proofs:
stirling_second(+non_negative_integer,+non_negative_integer,
-non_negative_integer) - zero_or_one

---

partition_number/2

Computes the partition number of a non-negative integer.

Compilation flags:
static

Template:
partition_number(N,PartitionNumber)
Mode and number of proofs:
partition_number(+non_negative_integer,-non_negative_integer) - one

---

binomial/3

Computes the binomial coefficient. `N` must be greater than or equal to `K` (fails otherwise).

Compilation flags:
`static`

Template:
`binomial(N,K,Binomial)`
Mode and number of proofs:
`binomial(+non_negative_integer,+non_negative_integer,-non_negative_integer)` - `zero_or_one`

---

multinomial/2

Computes the multinomial coefficient for a list of non-negative integers.

Compilation flags:
`static`

Template:
`multinomial(Integers,Multinomial)`
Mode and number of proofs:
`multinomial(+list(non_negative_integer),-non_negative_integer)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.339.15 number

Number data type predicates.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura

Version: 1:15:0
Date: 2026-02-11

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *atomic*

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 check/1 depth/2 ground/1 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 subsumes/2 subterm/2 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
  - *approximately_equal/2*
  - *approximately_equal/3*
  - *essentially_equal/3*
  - *tolerance_equal/4*
  - *=~= / 2*
- *Protected predicates*
- *Private predicates*
- *Operators*
  - *op(700,xfx,=~=)*

Public predicates

approximately_equal/2

Compares two numbers for approximate equality given the `epsilon` arithmetic constant value using the de facto standard formula `abs(Number1 - Number2) =< max(abs(Number1), abs(Number2)) * epsilon`. No type-checking.

Compilation flags:
`static`

Template:
`approximately_equal(Number1,Number2)`
Mode and number of proofs:
`approximately_equal(+number,+number)` - `zero_or_one`

approximately_equal/3

Compares two numbers for approximate equality given a user-defined epsilon value using the de facto standard formula `abs(Number1 - Number2) =< max(abs(Number1), abs(Number2)) * Epsilon`. No type-checking.

Compilation flags:

```
static
```

Template:

```
approximately_equal(Number1,Number2,Epsilon)
```

Mode and number of proofs:

```
approximately_equal(+number,+number,+number) - zero_or_one
```

Remarks:

- Epsilon range: Epsilon should be the `epsilon` arithmetic constant value or a small multiple of it. Only use a larger value if a greater error is expected.
- Comparison with essential equality: For the same epsilon value, approximate equality is weaker requirement than essential equality.

---

essentially_equal/3

Compares two numbers for essential equality given an epsilon value using the de facto standard formula abs(Number1 - Number2) =< min(abs(Number1), abs(Number2)) * Epsilon. No type-checking.

Compilation flags:

```
static
```

Template:

```
essentially_equal(Number1,Number2,Epsilon)
```

Mode and number of proofs:

```
essentially_equal(+number,+number,+number) - zero_or_one
```

Remarks:

- Comparison with approximate equality: For the same epsilon value, essential equality is a stronger requirement than approximate equality.

---

tolerance_equal/4

Compares two numbers for close equality given relative and absolute tolerances using the de facto standard formula abs(Number1 - Number2) =< max(RelativeTolerance * max(abs(Number1), abs(Number2)), AbsoluteTolerance). No type-checking.

Compilation flags:

```
static
```

Template:

```
tolerance_equal(Number1,Number2,RelativeTolerance,AbsoluteTolerance)
```

Mode and number of proofs:

`tolerance_equal(+number,+number,+number,+number)` - `zero_or_one`

`=~= / 2`

Compares two floats (or lists of floats) for approximate equality using 100*epsilon for the absolute error and, if that fails, 99.999% accuracy for the relative error. Note that these precision values may not be adequate for all cases. No type-checking.

Compilation flags:

`static`

Template:

`=~=(Float1,Float2)`

Mode and number of proofs:

`=~=(+number,+number)` - `zero_or_one`

`=~=(+list(number),+list(number))` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

`op(700,xfx,=~=)`

Scope:

`public`

object

1.339.16 `numberlist`

List of numbers predicates.

Availability:

`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:20:0
Date: 2026-05-06

Compilation flags:

`static, context_switching_calls`

Implements:

`public` *numberlistp*

Extends:

`public` *list*

Uses:

list

Remarks:

(none)

Inherited public predicates:

(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 append/2 append/3 as_difflist/2 average/2 chebyshev_distance/3 chebyshev_norm/2 check/1 delete/3 delete_matches/3 depth/2 drop/3 empty/1 euclidean_distance/3 euclidean_norm/2 flatten/2 ground/1 hamming_distance/3 keysort/2 last/2 least_common_multiple/2 length/2 linear_regression/4 manhattan_distance/3 manhattan_norm/2 max/2 median/2 member/2 memberchk/2 min/2 min_max/3 minkowski_distance/4 modes/2 msort/2 msort/3 new/1 nextto/3 normalize_range/2 normalize_range/4 normalize_scalar/2 normalize_unit/2 nth0/3 nth0/4 nth1/3 nth1/4 numbervars/1 numbervars/3 occurrences/2 occurrences/3 occurs/2 partition/5 permutation/2 prefix/2 prefix/3 product/2 proper_prefix/2 proper_prefix/3 proper_suffix/2 proper_suffix/3 remove_duplicates/2 rescale/3 reverse/2 same_length/2 same_length/3 scalar_product/3 select/3 select/4 selectchk/3 selectchk/4 sequential_occurrences/2 sequential_occurrences/3 singletons/2 softmax/2 softmax/3 sort/2 sort/3 sort/4 split/4 sublist/2 subsequence/3 subsequence/4 substitute/4 subsumes/2 subterm/2 subtract/3 suffix/2 suffix/3 sum/2 take/3 take/4 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

list, *list(Type)*, *varlist*, *difflist*

protocol

1.339.17 `numberlistp`

List of numbers protocol.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:13:0
Date: 2026-05-06

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *min/2*
  - *max/2*
  - *min_max/3*
  - *product/2*
  - *sum/2*
  - *average/2*
  - *median/2*
  - *modes/2*
  - *euclidean_norm/2*
  - *chebyshev_norm/2*
  - *manhattan_norm/2*
  - *euclidean_distance/3*
  - *chebyshev_distance/3*
  - *manhattan_distance/3*

- minkowski_distance/4
- scalar_product/3
- normalize_range/2
- normalize_range/4
- normalize_unit/2
- normalize_scalar/2
- rescale/3
- least_common_multiple/2
- softmax/2
- softmax/3
- linear_regression/4

- Protected predicates
- Private predicates
- Operators

Public predicates

min/2

Determines the minimum value in a list using arithmetic order. Fails if the list is empty.

Compilation flags:
`static`

Template:
`min(List,Minimum)`

Mode and number of proofs:
`min(+list(number),-number)` - `zero_or_one`

---

max/2

Determines the list maximum value using arithmetic order. Fails if the list is empty.

Compilation flags:
`static`

Template:
`max(List,Maximum)`

Mode and number of proofs:
`max(+list(number),-number)` - `zero_or_one`

---

min_max/3

Determines the minimum and maximum values in a list using arithmetic order. Fails if the list is empty.

Compilation flags:
`static`

Template:
`min_max(List,Minimum,Maximum)`
Mode and number of proofs:
`min_max(+list(number),-number,-number)` - `zero_or_one`

---

product/2

Calculates the product of all list numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`product(List,Product)`
Mode and number of proofs:
`product(+list(number),-number)` - `zero_or_one`

---

sum/2

Calculates the sum of all list numbers. Returns the integer zero if the list is empty.

Compilation flags:
`static`

Template:
`sum(List,Sum)`
Mode and number of proofs:
`sum(+list(number),-number)` - `one`

---

average/2

Calculates the average (i.e., arithmetic mean) of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`average(List,Average)`
Mode and number of proofs:
`average(+list(number),-float)` - `zero_or_one`

---

median/2

Calculates the median of a list of numbers. Fails if the list is empty.

Compilation flags:
`static`

Template:
`median(List,Median)`
Mode and number of proofs:
`median(+list(number),-float)` - `zero_or_one`

---

modes/2

Returns the list of modes of a list of numbers in ascending order. Fails if the list is empty.

Compilation flags:
`static`

Template:
`modes(List,Modes)`
Mode and number of proofs:
`modes(+list(number),-list(number))` - `zero_or_one`

---

euclidean_norm/2

Calculates the Euclidean norm of a list of numbers. Fails if the list is empty.

Compilation flags:
```
static
```

Template:
```
euclidean_norm(List,Norm)
```
Mode and number of proofs:
```
euclidean_norm(+list(number),-float) - zero_or_one
```

---

chebyshev_norm/2

Calculates the Chebyshev norm of a list of numbers. Fails if the list is empty.

Compilation flags:
```
static
```

Template:
```
chebyshev_norm(List,Norm)
```
Mode and number of proofs:
```
chebyshev_norm(+list(integer),-integer) - zero_or_one
chebyshev_norm(+list(float),-float) - zero_or_one
```

---

manhattan_norm/2

Calculates the Manhattan norm of a list of numbers. Fails if the list is empty.

Compilation flags:
```
static
```

Template:
```
manhattan_norm(List,Norm)
```
Mode and number of proofs:
```
manhattan_norm(+list(integer),-integer) - zero_or_one
manhattan_norm(+list(float),-float) - zero_or_one
```

---

euclidean_distance/3

Calculates the Euclidean distance between two lists of numbers. Fails if the two lists are empty or not of the same length.

Compilation flags:

```
static
```

Template:

```
euclidean_distance(List1,List2,Distance)
```

Mode and number of proofs:

```
euclidean_distance(+list(number),+list(number),-float) - zero_or_one
```

---

chebyshev_distance/3

Calculates the Chebyshev distance between two lists of numbers. Fails if the two lists are empty or not of the same length.

Compilation flags:

```
static
```

Template:

```
chebyshev_distance(List1,List2,Distance)
```

Mode and number of proofs:

```
chebyshev_distance(+list(integer),+list(integer),-integer) - zero_or_one
chebyshev_distance(+list(float),+list(float),-float) - zero_or_one
```

---

manhattan_distance/3

Calculates the Manhattan distance between two lists of numbers. Fails if the two lists are empty or not of the same length.

Compilation flags:

```
static
```

Template:

```
manhattan_distance(List1,List2,Distance)
```

Mode and number of proofs:

```
manhattan_distance(+list(integer),+list(integer),-integer) - zero_or_one
manhattan_distance(+list(float),+list(float),-float) - zero_or_one
```

---

minkowski_distance/4

Calculates the Minkowski distance between two lists of numbers given a positive power. Fails if the two lists are empty or not of the same length.

Compilation flags:

```
static
```

Template:

```
minkowski_distance(List1,List2,Power,Distance)
```

Mode and number of proofs:

```
minkowski_distance(+list(number),+list(number),+number,-float) - zero_or_one
```

---

scalar_product/3

Calculates the scalar product of two lists of numbers. Fails if the two lists are empty or not of the same length.

Compilation flags:

```
static
```

Template:

```
scalar_product(List1,List2,Product)
```

Mode and number of proofs:

```
scalar_product(+list(integer),+list(integer),-integer) - zero_or_one
scalar_product(+list(float),+list(float),-float) - zero_or_one
```

---

normalize_range/2

Normalizes a list of numbers into the `[0.0,1.0]` range. Caller must handle arithmetic exceptions if the input list if not normalizable.

Compilation flags:

```
static
```

Template:

```
normalize_range(List,NormalizedList)
```

Mode and number of proofs:

```
normalize_range(+list(number),-list(float)) - one
```

---

normalize_range/4

Normalizes a list of numbers into the given range. Caller must handle arithmetic exceptions if the input list if not normalizable.

Compilation flags:

```
static
```

Template:

```
normalize_range(List,Minimum,Maximum,NormalizedList)
```

Mode and number of proofs:

```
normalize_range(+list(number),+number,+number,-list(float)) - one
```

---

normalize_unit/2

Normalizes a list of numbers returning its unit vector (i.e., a list with Euclidean norm equal to one). Caller must handle arithmetic exceptions if the input list if not normalizable.

Compilation flags:

```
static
```

Template:

```
normalize_unit(List,NormalizedList)
```

Mode and number of proofs:

```
normalize_unit(+list(number),-list(float)) - one
```

---

normalize_scalar/2

Normalizes a list of numbers such that the sum of all numbers is equal to one. Caller must handle arithmetic exceptions if the input list if not normalizable.

Compilation flags:

```
static
```

Template:

```
normalize_scalar(List,NormalizedList)
```

Mode and number of proofs:

```
normalize_scalar(+list(number),-list(float)) - one
```

---

rescale/3

Rescales all numbers in a list by the given factor.

Compilation flags:
```
static
```

Template:
```
rescale(List,Factor,RescaledList)
```
Mode and number of proofs:
```
rescale(+list(integer),+integer,-list(integer)) - one
rescale(+list(number),+float,-list(float)) - one
```

---

least_common_multiple/2

Computes the least common multiple of a list of two or more positive integers. Fails if the list is empty or contains a single element. Fails also if any of the elements is zero. May require backend support for unbound integer arithmetic.

Compilation flags:
```
static
```

Template:
```
least_common_multiple(Integers,LeastCommonMultiple)
```
Mode and number of proofs:
```
least_common_multiple(+list(positive_integer),-positive_integer) - zero_or_one
```

---

softmax/2

Computes the softmax of a list of floats, returning a probability distribution. The implementation is numerically stable.

Compilation flags:
```
static
```

Template:
```
softmax(Floats,Softmax)
```
Mode and number of proofs:
```
softmax(+list(float),-list(float)) - one
```

---

softmax/3

Computes the softmax of a list of floats with the given temperature, returning a probability distribution. The implementation is numerically stable.

Compilation flags:

```
static
```

Template:

```
softmax(Floats,Temperature,Softmax)
```

Mode and number of proofs:

```
softmax(+list(float),+positive_float,-list(float)) - one
```

Remarks:

- `Temperature > 1.0`: Makes the distribution more uniform.
- `Temperature < 1.0`: Makes the distribution more concentrated on the largest values.
- `Temperature = 1.0`: Standard softmax behavior.

---

linear_regression/4

Computes the simple linear regression for the given lists of X and Y coordinates, returning the slope and the intercept. Fails if the lists have less than two elements, are not of the same length, or if all X values are the same.

Compilation flags:

```
static
```

Template:

```
linear_regression(Xs,Ys,Slope,Intercept)
```

Mode and number of proofs:

```
linear_regression(+list(number),+list(number),-float,-float) - zero_or_one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

numberlist, *listp*, *varlistp*

object

1.339.18 `pairs`

Useful predicates over lists of pairs (key-value terms).

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 2:1:1
Date: 2023-11-21

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:

- Usage: This object can be loaded independently of other entities in the `types` library by using the goal `logtalk_load(types(pairs))`.

Inherited public predicates:
(none)

- *Public predicates*
  - *keys_values/3*
  - *keys/2*
  - *key/2*
  - *values/2*
  - *value/3*
  - *transpose/2*
  - *group_sorted_by_key/2*
  - *group_consecutive_by_key/2*
  - *group_by_key/2*
  - *map/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

keys_values/3

Converts between a list of pairs and lists of keys and values. When converting to pairs, this predicate fails if the list of keys and the list of values have different lengths.

Compilation flags:

```
static
```

Template:

```
keys_values(Pairs,Keys,Values)
```

Mode and number of proofs:

```
keys_values(+list(pair),-list,-list) - one
keys_values(-list(pair),+list,+list) - zero_or_one
```

---

keys/2

Returns a list of keys from a list of pairs.

Compilation flags:

```
static
```

Template:

```
keys(Pairs,Keys)
```

Mode and number of proofs:

```
keys(+list(pair),-list) - one
```

---

key/2

Enumerates by backtracking all keys from a list of pairs.

Compilation flags:

```
static
```

Template:

```
key(Pairs,Key)
```

Mode and number of proofs:

```
key(+list(pair),-term) - zero_or_more
```

---

values/2

Returns a list of values from a list of pairs.

Compilation flags:
```
static
```

Template:
```
values(Pairs,Values)
```
Mode and number of proofs:
```
values(+list(pair),-list) - one
```

---

value/3

Returns a value addressed by the given path (a key or a list of keys in the case of nested list of pairs). Fails if path does not exist.

Compilation flags:
```
static
```

Template:
```
value(Pairs,Path,Value)
```
Mode and number of proofs:
```
value(+list(pair),+term,-term) - zero_or_one
value(+list(pair),+list,-term) - zero_or_one
```

---

transpose/2

Transposes a list of pairs by swapping each pair key and value. The relative order of the list elements is kept.

Compilation flags:
```
static
```

Template:
```
transpose(Pairs,TransposedPairs)
```
Mode and number of proofs:
```
transpose(+list(pair),-list(pair)) - one
```

---

group_sorted_by_key/2

Groups pairs by key by sorting them and then constructing new pairs by grouping all values for a given key in a list. Keys are compared using equality. Relative order of values per key is kept. Resulting list of pairs is sorted by key.

Compilation flags:

```
static
```

Template:

```
group_sorted_by_key(Pairs,Groups)
```

Mode and number of proofs:

```
group_sorted_by_key(+list(pair),-list(pair)) - one
```

---

group_consecutive_by_key/2

Groups pairs by constructing new pairs by grouping all values for consecutive key in a list. Keys are compared using equality. The relative order of the values for the same key is kept.

Compilation flags:

```
static
```

Template:

```
group_consecutive_by_key(Pairs,Groups)
```

Mode and number of proofs:

```
group_consecutive_by_key(+list(pair),-list(pair)) - one
```

---

group_by_key/2

Same as the `group_sorted_by_key/2` predicate. Deprecated.

Compilation flags:

```
static
```

Template:

```
group_by_key(Pairs,Groups)
```

Mode and number of proofs:

```
group_by_key(+list(pair),-list(pair)) - one
```

`map/3`

Maps a list into pairs using a closure that applies to each list element to compute its key.

Compilation flags:
`static`

Template:
`map(Closure,List,Pairs)`
Meta-predicate template:
`map(2,*,*)`
Mode and number of proofs:
`map(@callable,+list,-list(pair))` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.339.19 `term`

Term utility predicates.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:11:0
Date: 2022-05-13

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *termp*
Aliases:
`termp` `variables/2` as `vars/2`

Remarks:
(none)

Inherited public predicates:

(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 check/1 depth/2 ground/1 new/1 numbervars/1 numbervars/3 occurs/2 singletons/2 subsumes/2 subterm/2 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.339.20 `termp`

Term utility predicates protocol.

Availability:

`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:35:0
Date: 2022-05-13

Compilation flags:

`static`

Extends:

`public` *comparingp*

Remarks:

(none)

Inherited public predicates:

(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2

- *Public predicates*
  - *depth/2*
  - *ground/1*
  - *new/1*
  - *occurs/2*
  - *subsumes/2*
  - *subterm/2*
  - *valid/1*
  - *check/1*
  - *variant/2*
  - *variables/2*
  - *singletons/2*
  - *numbervars/3*
  - *numbervars/1*
  - *varnumbers/3*
  - *varnumbers/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

depth/2

True if the depth of `Term` is `Depth`. The depth of atomic terms is zero; the depth of a compound term is one plus the maximum depth of its sub-terms.

Compilation flags:
`static`

Template:
`depth(Term,Depth)`

Mode and number of proofs:
`depth(@term,?integer)` - `zero_or_one`

---

ground/1

True if the argument is ground. Deprecated. Use the `ground/1` standard predicate instead.

Compilation flags:
`static`

Template:
`ground(Term)`

Mode and number of proofs:

```
ground(@term) - zero_or_one
```

---

new/1

Creates a new term instance (if meaningful).

Compilation flags:

```
static
```

Template:

```
new(Term)
```

Mode and number of proofs:

```
new(-nonvar) - zero_or_one
```

---

occurs/2

True if the variable occurs in the term.

Compilation flags:

```
static
```

Template:

```
occurs(Variable,Term)
```

Mode and number of proofs:

```
occurs(@var,@term) - zero_or_one
```

---

subsumes/2

The first term subsumes the second term. Deprecated. Use the `subsumes_term/2` standard predicate instead.

Compilation flags:

```
static
```

Template:

```
subsumes(General,Specific)
```

Mode and number of proofs:

```
subsumes(@term,@term) - zero_or_one
```

---

subterm/2

The first term is a subterm of the second term.

Compilation flags:
`static`

Template:
`subterm(Subterm,Term)`
Mode and number of proofs:
`subterm(?term,+term)` - `zero_or_more`

---

valid/1

Term is valid.

Compilation flags:
`static`

Template:
`valid(Term)`
Mode and number of proofs:
`valid(@nonvar)` - `zero_or_one`

---

check/1

Checks if a term is valid. Throws an exception if the term is not valid.

Compilation flags:
`static`

Template:
`check(Term)`
Mode and number of proofs:
`check(@nonvar)` - `one`

---

variant/2

Each term is a variant of the other (i.e., they are structurally equivalent).

Compilation flags:
```
static
```

Template:
```
variant(Term1,Term2)
```
Mode and number of proofs:
```
variant(@term,@term) - zero_or_one
```

---

variables/2

Returns a list of all term variables (ordered as found when doing a depth-first, left-to-right traversal of `Term`). Deprecated. Use the standard `term_variables/2` predicate instead.

Compilation flags:
```
static
```

Template:
```
variables(Term,List)
```
Mode and number of proofs:
```
variables(@term,-list) - one
```

---

singletons/2

Returns a list of all term singleton variables (ordered as found when doing a depth-first, left-to-right traversal of `Term`).

Compilation flags:
```
static
```

Template:
```
singletons(Term,Singletons)
```
Mode and number of proofs:
```
singletons(@term,-list) - one
```

---

numbervars/3

Grounds a term by replacing all variables with `'$VAR'(N)` terms with `N` starting at `From`. The `Next` argument is unified with the next value for `N` after binding all variables.

Compilation flags:

`static`

Template:

`numbervars(Term,From,Next)`

Mode and number of proofs:

`numbervars(?term,+integer,?integer)` - `zero_or_one`

---

numbervars/1

Grounds a term by replacing all variables with `'$VAR'(N)` terms with `N` starting at `0`.

Compilation flags:

`static`

Template:

`numbervars(Term)`

Mode and number of proofs:

`numbervars(?term)` - `zero_or_one`

---

varnumbers/3

Replaces all `'$VAR'(N)` sub-terms in a term with fresh variables for all values of `N` grater or equal to `From`. Variables in `Term` are shared with `Copy`.

Compilation flags:

`static`

Template:

`varnumbers(Term,From,Copy)`

Mode and number of proofs:

`varnumbers(@term,+integer,?term)` - `zero_or_one`

---

varnumbers/2

Replaces all '$VAR'(N) sub-terms in a term with fresh variables for all values of N grater or equal to 0. Variables in Term are shared with Copy.

Compilation flags:
```
static
```

Template:
```
varnumbers(Term,Copy)
```
Mode and number of proofs:
```
varnumbers(@term,?term) - zero_or_one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

term

object

1.339.21 type

Type checking predicates. User extensible. New types can be defined by adding clauses for the type/1 and check/2 multifile predicates.

Availability:
```
logtalk_load(types(loader))
```

Author: Paulo Moura
Version: 2:9:2
Date: 2026-10-29

Compilation flags:
```
static, context_switching_calls, complements(restrict)
```

Uses:
list

Remarks:

- Logtalk specific types: `entity`, `object`, `protocol`, `category`, `entity_identifier`, `object_identifier`, `protocol_identifier`, `category_identifier`, `event`, `predicate`.
- Prolog module related types (when the backend compiler supports modules): `module`, `module_identifier`, `qualified_callable`.
- Prolog base types: `term`, `var`, `nonvar`, `atomic`, `atom`, `number`, `integer`, `float`, `compound`, `callable`, `ground`.
- Atom derived types: `non_quoted_atom`, `non_empty_atom`, `boolean`, `character`, `in_character`, `char`, `operator_specifier`, `hex_char`.
- Atom derived parametric types: `atom(CharSet)`, `atom(CharSet,Length)`, `non_empty_atom(CharSet)`, `character(CharSet)`, `in_character(CharSet)`, `char(CharSet)`.
- Number derived types: `positive_number`, `negative_number`, `non_positive_number`, `non_negative_number`.
- Float derived types: `positive_float`, `negative_float`, `non_positive_float`, `non_negative_float`, `probability`, `open_probability`.
- Integer derived types: `positive_integer`, `negative_integer`, `non_positive_integer`, `non_negative_integer`, `byte`, `in_byte`, `character_code`, `in_character_code`, `code`, `operator_priority`, `hex_code`.
- Integer derived parametric types: `character_code(CharSet)`, `in_character_code(CharSet)`, `code(CharSet)`.
- List types (compound derived types): `list`, `non_empty_list`, `partial_list`, `list_or_partial_list`, `list(Type)`, `list(Type,Length)`, `list(Type,Min,Max)`, `list(Type,Length,Min,Max)`, `non_empty_list(Type)`, `codes`, `codes(CharSet)`, `chars`, `chars(CharSet)`.
- Difference list types (compound derived types): `difference_list`, `difference_list(Type)`.
- Other compound derived types: `compound(Name,Types)`, `predicate_indicator`, `non_terminal_indicator`, `predicate_or_non_terminal_indicator`, `clause`, `grammar_rule`, `pair`, `pair(KeyType,ValueType)`, `cyclic`, `acyclic`.
- Stream types: `stream`, `stream_or_alias`, `stream(Property)`, `stream_or_alias(Property)`.
- Other types: `text`, `text(CharSet)`, `Object::Closure`, `between(Type,Lower,Upper)`, `property(Type,LambdaExpression)`, `one_of(Type,Set)`, `var_or(Type)`, `ground(Type)`, `types(Types)`, `constrain(Type,Closure)`, `type`.
- Type `predicate` notes: This type is used to check for an object public predicate specified as `Object::Functor/Arity`.
- Type `boolean` notes: The two value of this type are the atoms `true` and `false`.
- Stream types notes: In the case of the `stream(Property)` and `stream_or_alias(Property)` types, Property must be a valid stream property.
- Type `order` notes: The three possible values of this type are the single character atoms `<`, `=`, and `>`.
- Type `character_code` notes: This type takes into account Unicode support by the backend compiler. When Unicode is supported, it distinguishes between BMP and full support. When Unicode is not supported, it assumes a byte representation for characters.
- Types `text` and `text(CharSet)` notes: These types allow text to be represented using atoms, character lists, or character code lists.
- Type `Object::Closure` notes: Allows calling a public object predicate for type-checking. The predicate should provide `valid/2` predicate semantics and assume called with a bound argument. The `Closure` closure is extended with a single argument, the value to be checked.
- Type `compound(Name,Types)` notes: This type verifies that a compound term have the given `Name` and its arguments conform to `Types`.
- Type `between(Type, Lower, Upper)` notes: The type argument allows distinguishing between numbers and other types. It also allows choosing between mixed integer/float comparisons and strict float or integer comparisons. The term is type-checked before testing for interval membership.
- Type `property(Type, Lambda)` notes: Verifies that `Term` satisfies a property described using a lambda expression of the form `[Parameter]>>Goal`. The lambda expression is applied in the context of `user`. The term is type-checked before calling the goal.
- Type `one_of(Type, Set)` notes: For checking if a given term is an element of a set. The set is represented using a list. The term is type-checked before testing for set membership.

- Type `var_or(Type)` notes: Allows checking if a term is either a variable or a valid value of the given type.
- Type `ground(Type)` notes: Allows checking if a term is ground and a valid value of the given type.
- Type `types(Types)` notes: Allows checking if a term is a valid value for one of the types in a list of types.
- Type `constrain(Type,Closure)` notes: Allows checking if a term is a valid value for the given type and satisfies the given closure.
- Type `type` notes: Allows checking if a term is a valid type.
- Type `qualified_callable` notes: Allows checking if a term is a possibly module-qualified callable term. When the term is qualified, it also checks that the qualification modules are type correct. When the term is not qualified, its semantics are the same as the callable type.
- Design choices: The main predicates are `valid/2` and `check/3`. These are defined using the predicate `check/2`. Defining clauses for `check/2` instead of `valid/2` gives the user full control of exception terms without requiring an additional predicate.
- Error context: The built-in execution-context method `context/1` can be used to provide the calling context for errors when using the predicate `check/3`.
- Registering new types: New types can be registered by defining clauses for the `type/1` and `check/2` multifile predicates. Clauses for both predicates must have a bound first argument to avoid introducing spurious choice-points when type-checking terms.
- Meta-types: Meta-types are types that have one or more sub-type arguments. E.g. `var_or(Type)`. The sub-types of a meta-type can be enumerated by defining a clause for the `meta_type/3` multifile predicate.
- Character sets: When testing character or character code based terms (e.g., atom), it is possible to choose a character set (`ascii_identifier`, `ascii_printable`, `ascii_full`, `hexadecimal`, `byte`, `unicode_bmp`, or `unicode_full`) using the parameterizable types.
- Caveats: The type argument (and any type parameterization) to the predicates is not type-checked (or checked for consistency) for performance reasons.
- Unicode limitations: Currently, correct character/code type-checking is only ensured for SWI-Prolog and XVM as other backends do not provide support for querying a Unicode code point category.
- Random seed: When setting the random generator seed using the `set_seed/1` predicate inherited from the `arbitrary` category, the seed must be a valid value for the portable random generator provided by the `random` library.

Inherited public predicates:

arbitrary/1 arbitrary/2 edge_case/2 get_seed/1 max_size/1 mutation/3 set_seed/1 shrink/3 shrink_sequence/3 shrinker/1

- *Public predicates*
  - *type/1*
  - *meta_type/3*
  - *valid/2*
  - *check/3*
  - *check/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

type/1

Table of defined types. A new type can be registered by defining a clause for this predicate and adding a clause for the check/2 multifile predicate.

Compilation flags:

```
static, multifile
```

Template:

```
type(Type)
```

Mode and number of proofs:

```
type(?callable) - zero_or_more
```

---

meta_type/3

Table of defined meta-types. A registered type that is a meta-type can be described by defining a clause for this predicate to enumerate its sub-types and optional values in case of a single sub-type.

Compilation flags:

```
static, multifile
```

Template:

```
meta_type(MetaType,SubTypes,Values)
```

Mode and number of proofs:

```
meta_type(?callable,-list,-list) - zero_or_more
```

---

valid/2

True if the given term is of the specified type. Fails otherwise.

Compilation flags:

```
static
```

Template:

```
valid(Type,Term)
```

Mode and number of proofs:

```
valid(@callable,@term) - zero_or_one
```

---

check/3

True if the given term is of the specified type. Throws an error otherwise using the format `error(Error, Context)`. For the possible values of `Error` see the `check/2` predicate.

Compilation flags:

`static`

Template:

`check(Type,Term,Context)`

Mode and number of proofs:

`check(@callable,@term,@term)` - `one_or_error`

---

check/2

True if the given term is of the specified type. Throws an error otherwise. A new type can be added by defining a clause for this predicate and registering it by adding a clause for the `type/1` multifile predicate.

Compilation flags:

`static, multifile`

Template:

`check(Type,Term)`

Meta-predicate template:

`check(::,*)`

Mode and number of proofs:

`check(@callable,@term)` - `one_or_error`

Exceptions:

`Term` is not bound as required:

`instantiation_error`

`Term` is bound but not of the specified type:

`type_error(Type,Term)`

`Term` is the of the correct type but not in the specified domain:

`domain_error(Domain,Term)`

`Term` is the of the correct type and domain but the resource it represents does not exist:

`existence_error(Type,Term)`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

arbitrary, *os_types*, *either*, *maybe*

object

1.339.22 `varlist`

List of variables predicates.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 2:0:0
Date: 2020-05-11

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *varlistp*

Remarks:
(none)

Inherited public predicates:
append/3 *check/1* *delete/3* *empty/1* *flatten/2* *last/2* *length/2* *memberchk/2* *nextto/3* *nth0/3* *nth0/4* *nth1/3* *nth1/4* *permutation/2* *prefix/2* *remove_duplicates/2* *reverse/2* *same_length/2* *select/3* *sublist/2* *subtract/3* *suffix/2* *valid/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

list, *list(Type)*, *numberlist*, *difflist*

protocol

1.339.23 `varlistp`

List of variables protocol.

Availability:
`logtalk_load(types(loader))`

Author: Paulo Moura
Version: 1:3:0
Date: 2022-09-19

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *append/3*
  - *delete/3*
  - *empty/1*
  - *flatten/2*

- last/2
- length/2
- memberchk/2
- nextto/3
- nth0/3
- nth0/4
- nth1/3
- nth1/4
- permutation/2
- prefix/2
- remove_duplicates/2
- reverse/2
- same_length/2
- select/3
- sublist/2
- subtract/3
- suffix/2
- valid/1
- check/1

- Protected predicates
- Private predicates
- Operators

Public predicates

append/3

Appends two lists.

Compilation flags:

```
static
```

Template:

```
append(List1,List2,List)
```

Mode and number of proofs:

```
append(?list,?list,?list) - zero_or_more
```

delete/3

Deletes from a list all occurrences of an element returning the list of remaining elements.

Compilation flags:
```
static
```

Template:
```
delete(List,Element,Remaining)
```
Mode and number of proofs:
```
delete(@list,@term,?list) - one
```

---

empty/1

True if the argument is an empty list.

Compilation flags:
```
static
```

Template:
```
empty(List)
```
Mode and number of proofs:
```
empty(@list) - zero_or_one
```

---

flatten/2

Flattens a list of lists into a list.

Compilation flags:
```
static
```

Template:
```
flatten(List,Flatted)
```
Mode and number of proofs:
```
flatten(@list,-list) - one
```

---

last/2

List last element (if it exists).

Compilation flags:
`static`

Template:
`last(List,Last)`
Mode and number of proofs:
`last(@list,@var)` - `zero_or_one`

---

length/2

List length.

Compilation flags:
`static`

Template:
`length(List,Length)`
Mode and number of proofs:
`length(@list,?integer)` - `zero_or_one`

---

memberchk/2

Checks if a variable is a member of a list.

Compilation flags:
`static`

Template:
`memberchk(Element,List)`
Mode and number of proofs:
`memberchk(@var,@list)` - `zero_or_one`

---

nextto/3

X and Y are consecutive elements in List.

Compilation flags:

```
static
```

Template:

```
nextto(X,Y,List)
```

Mode and number of proofs:

```
nextto(@var,@var,?list) - zero_or_more
```

---

nth0/3

Nth element of a list (counting from zero).

Compilation flags:

```
static
```

Template:

```
nth0(Nth,List,Element)
```

Mode and number of proofs:

```
nth0(?integer,+list,@var) - zero_or_more
```

---

nth0/4

Nth element of a list (counting from zero). Rest is a list of all the other elements. Can be used to either select the nth element of List or to insert an element before the nth element in Rest.

Compilation flags:

```
static
```

Template:

```
nth0(Nth,List,Element,Rest)
```

Mode and number of proofs:

```
nth0(?integer,+list,@var,?list) - zero_or_more
```

---

nth1/3

Nth element of a list (counting from one).

Compilation flags:
`static`

Template:
`nth1(Nth,List,Element)`
Mode and number of proofs:
`nth1(?integer,+list,@var)` - `zero_or_more`

---

nth1/4

Nth element of a list (counting from zero). `Rest` is a list of all the other elements. Can be used to either select the nth element of `List` or to insert an element before the nth element in `Rest`.

Compilation flags:
`static`

Template:
`nth1(Nth,List,Element,Rest)`
Mode and number of proofs:
`nth1(?integer,+list,@var,?list)` - `zero_or_more`

---

permutation/2

The two lists are a permutation of the same list.

Compilation flags:
`static`

Template:
`permutation(List,Permutation)`
Mode and number of proofs:
`permutation(@list,@list)` - `zero_or_one`

---

prefix/2

`Prefix` is a prefix of `List`.

Compilation flags:
`static`

Template:
`prefix(Prefix,List)`
Mode and number of proofs:
`prefix(?list,@list)` - `zero_or_more`

---

remove_duplicates/2

Removes duplicated variables and keeping the left-most variable when repeated.

Compilation flags:
`static`

Template:
`remove_duplicates(List,Set)`
Mode and number of proofs:
`remove_duplicates(+list,-list)` - `one`

---

reverse/2

Reverses a list.

Compilation flags:
`static`

Template:
`reverse(List,Reversed)`
Mode and number of proofs:
`reverse(@list,?list)` - `zero_or_one`
`reverse(?list,@list)` - `zero_or_one`
`reverse(-list,-list)` - `one_or_more`

---

same_length/2

The two lists have the same length.

Compilation flags:
```
static
```

Template:
```
same_length(List1,List2)
```
Mode and number of proofs:
```
same_length(@list,?list) - zero_or_one
same_length(?list,@list) - zero_or_one
same_length(-list,-list) - one_or_more
```

---

select/3

Selects an element from a list, returning the list of remaining elements.

Compilation flags:
```
static
```

Template:
```
select(Element,List,Remaining)
```
Mode and number of proofs:
```
select(@var,?list,?list) - zero_or_more
```

---

sublist/2

The first list is a sublist of the second.

Compilation flags:
```
static
```

Template:
```
sublist(Sublist,List)
```
Mode and number of proofs:
```
sublist(?list,@list) - zero_or_more
```

---

subtract/3

Removes all elements in the second list from the first list, returning the list of remaining elements.

Compilation flags:
`static`

Template:
`subtract(List,Elements,Remaining)`
Mode and number of proofs:
`subtract(@list,@list,-list)` - `one`

---

suffix/2

`Suffix` is a suffix of `List`.

Compilation flags:
`static`

Template:
`suffix(Suffix,List)`
Mode and number of proofs:
`suffix(?list,@list)` - `zero_or_more`

---

valid/1

Term is a valid list of variables.

Compilation flags:
`static`

Template:
`valid(Term)`
Mode and number of proofs:
`valid(@nonvar)` - `zero_or_one`

`check/1`

Checks if a term is a valid list of variables. Throws an exception if the term is not valid.

Compilation flags:
`static`

Template:
`check(Term)`
Mode and number of proofs:
`check(@nonvar)` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

varlist, *listp*, *numberlistp*

1.340 tzif

object

1.340.1 `tzif`

Loader, per-zone cache and snapshot persistence support, and zone-aware UTC lookup predicates for TZif v1/v2/v3 files.

Availability:
`logtalk_load(tzif(loader))`

Author: Paulo Moura
Version: 1:1:1
Date: 2026-08-02

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *tzif_protocol*

Uses:

date
list
logtalk
os
reader
tzif_zone_ids

Remarks:

(none)

Inherited public predicates:

abbreviation/2 *abbreviation/3* *abbreviation/4* *cache/1* *cache_source/1* *cached_tzif/1* *clear_cache/0*
daylight_saving_time/2 *daylight_saving_time/3* *daylight_saving_time/4* *load/1* *load/2* *local_abbreviation/2*
local_abbreviation/3 *local_abbreviation/4* *local_abbreviation_reified/2* *local_abbreviation_reified/3*
local_abbreviation_reified/4 *local_abbreviation_with_resolution/3* *local_abbreviation_with_resolution/4*
local_abbreviation_with_resolution/5 *local_daylight_saving_time/2* *local_daylight_saving_time/3*
local_daylight_saving_time/4 *local_daylight_saving_time_reified/2* *local_daylight_saving_time_reified/3*
local_daylight_saving_time_reified/4 *local_daylight_saving_time_with_resolution/3*
local_daylight_saving_time_with_resolution/4 *local_daylight_saving_time_with_resolution/5* *local_offset/2* *local_offset/3*
local_offset/4 *local_offset_reified/2* *local_offset_reified/3* *local_offset_reified/4* *local_offset_with_resolution/3*
local_offset_with_resolution/4 *local_offset_with_resolution/5* *local_time_type/2* *local_time_type/3* *local_time_type/4*
local_time_type_reified/2 *local_time_type_reified/3* *local_time_type_reified/4* *local_time_type_with_resolution/3*
local_time_type_with_resolution/4 *local_time_type_with_resolution/5* *offset/2* *offset/3* *offset/4* *save/1* *save/2* *time_type/2*
time_type/3 *time_type/4* *zone/3* *zones/1* *zones/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - `cached_tzif_/2`
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`cached_tzif_/2`

Table holding the currently cached TZif terms keyed by zone identifier.

Compilation flags:

`dynamic`

Template:

```
cached_tzif_(Zone,TZif)
```

Mode and number of proofs:

```
cached_tzif_(?atom,?compound) - zero_or_more
```

Operators

(none)

protocol

1.340.2 `tzif_protocol`

Protocol for loading TZif data sets, persisting loaded terms, and answering zone-aware UTC-based offset, DST, and abbreviation queries.

Availability:

```
logtalk_load(tzif(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2026-06-15

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *load/1*
  - *load/2*
  - *cache/1*
  - *save/2*
  - *save/1*
  - *clear_cache/0*
  - *cache_source/1*
  - *cached_tzif/1*
  - *zone/3*
  - *zones/2*
  - *zones/1*

```
      - time_type/4
      - time_type/3
      - time_type/2
      - offset/4
      - offset/3
      - offset/2
      - daylight_saving_time/4
      - daylight_saving_time/3
      - daylight_saving_time/2
      - abbreviation/4
      - abbreviation/3
      - abbreviation/2
      - local_time_type/4
      - local_time_type/3
      - local_time_type/2
      - local_time_type_with_resolution/5
      - local_time_type_with_resolution/4
      - local_time_type_with_resolution/3
      - local_offset/4
      - local_offset/3
      - local_offset/2
      - local_offset_with_resolution/5
      - local_offset_with_resolution/4
      - local_offset_with_resolution/3
      - local_daylight_saving_time/4
      - local_daylight_saving_time/3
      - local_daylight_saving_time/2
      - local_daylight_saving_time_with_resolution/5
      - local_daylight_saving_time_with_resolution/4
      - local_daylight_saving_time_with_resolution/3
      - local_abbreviation/4
      - local_abbreviation/3
      - local_abbreviation/2
      - local_abbreviation_with_resolution/5
      - local_abbreviation_with_resolution/4
      - local_abbreviation_with_resolution/3
      - local_time_type_reified/4
      - local_time_type_reified/3
      - local_time_type_reified/2
      - local_offset_reified/4
      - local_offset_reified/3
      - local_offset_reified/2
      - local_daylight_saving_time_reified/4
      - local_daylight_saving_time_reified/3
      - local_daylight_saving_time_reified/2
      - local_abbreviation_reified/4
      - local_abbreviation_reified/3
```

    - *local_abbreviation_reified/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

load/1

Loads a TZif source given as `file(Path, ZoneId)`, `files(Root, Paths)`, `directory(Root)`, `stream(Stream, ZoneId)`, `bytes(Bytes, ZoneId)`, or `snapshot(File)` and caches the resulting per-zone `tzif(...)` terms, replacing cached entries with matching zone identifiers. For `directory(Root)` sources, regular files whose relative paths are not recognized zone identifiers are ignored, allowing system zoneinfo trees that contain metadata files such as `leapseconds`. Zone identifiers are validated against bundled IANA TZDB 2026a canonical names plus backward-compatible aliases.

Compilation flags:

`static`

Template:

`load(Source)`

Mode and number of proofs:

`load(+compound)` - `zero_or_one`

---

load/2

Loads a TZif source given as `file(Path, ZoneId)`, `files(Root, Paths)`, `directory(Root)`, `stream(Stream, ZoneId)`, `bytes(Bytes, ZoneId)`, or `snapshot(File)` into a list of per-zone `tzif(...)` compound terms without caching them. For `directory(Root)` sources, regular files whose relative paths are not recognized zone identifiers are ignored, allowing system zoneinfo trees that contain metadata files such as `leapseconds`. Zone identifiers are validated against bundled IANA TZDB 2026a canonical names plus backward-compatible aliases.

Compilation flags:

`static`

Template:

`load(Source,TZifs)`

Mode and number of proofs:

`load(+compound,-list(compound))` - `zero_or_one`

---

cache/1

Caches one or more loaded per-zone `tzif(...)` compound terms, replacing cached entries with matching zone identifiers.

Compilation flags:
`static`

Template:
`cache(TZifs)`
Mode and number of proofs:
`cache(+list(compound))` - `one`

---

save/2

Saves a list of per-zone `tzif(...)` terms to a plain Prolog snapshot file, writing one serialized term per line.

Compilation flags:
`static`

Template:
`save(TZifs,File)`
Mode and number of proofs:
`save(+list(compound),+atom)` - `one`

---

save/1

Saves all cached `tzif(...)` terms to a plain Prolog snapshot file.

Compilation flags:
`static`

Template:
`save(File)`
Mode and number of proofs:
`save(+atom)` - `one_or_error`

Exceptions:
No TZif terms are cached:
`existence_error(tzif_cache,tzif)`
`File` is a variable:
`instantiation_error`
`File` is neither a variable nor a valid file name atom:
`domain_error(file_name,File)`

---

clear_cache/0

Clears all cached TZif terms.

Compilation flags:
`static`

Mode and number of proofs:
`clear_cache` - `one`

---

cache_source/1

Enumerates the source terms recorded in the cached TZif terms.

Compilation flags:
`static`

Template:
`cache_source(Source)`
Mode and number of proofs:
`cache_source(-compound)` - `zero_or_more`

---

cached_tzif/1

Enumerates the cached per-zone `tzif(...)` terms.

Compilation flags:
`static`

Template:
`cached_tzif(TZif)`
Mode and number of proofs:
`cached_tzif(-compound)` - `zero_or_more`

---

zone/3

Returns the loaded zone identifier and its nested parsed zone-data term from a per-zone `tzif(...)` term.

Compilation flags:
`static`

Template:

```
zone(TZif,Zone,ZoneData)
```

Mode and number of proofs:

```
zone(+compound,?atom,-compound) - zero_or_one
```

---

zones/2

Returns the list of zone identifiers loaded in a list of per-zone `tzif(...)` terms.

Compilation flags:

```
static
```

Template:

```
zones(TZifs,Zones)
```

Mode and number of proofs:

```
zones(+list(compound),-list(atom)) - one
```

---

zones/1

Returns the list of zone identifiers loaded in the cached `tzif(...)` terms.

Compilation flags:

```
static
```

Template:

```
zones(Zones)
```

Mode and number of proofs:

```
zones(-list(atom)) - one_or_error
```

Exceptions:

No TZif terms are cached:

```
existence_error(tzif_cache,tzif)
```

---

time_type/4

Returns the applicable local time type for a loaded zone and a UTC instant given either as Unix seconds or as a `date_time/6` term.

Compilation flags:

```
static
```

Template:

```
time_type(TZif,Zone,UTC,TimeType)
```

Mode and number of proofs:
`time_type(+compound,+atom,+types([integer,compound]),-compound)` - `zero_or_one`

---

time_type/3

Returns the applicable local time type for a zone in the cached TZif terms.

Compilation flags:
`static`

Template:
`time_type(Zone,UTC,TimeType)`
Mode and number of proofs:
`time_type(+atom,+types([integer,compound]),-compound)` - `one_or_error`

Exceptions:
`Zone` is not present in the cached TZif terms:
`existence_error(time_zone,Zone)`

---

time_type/2

Cached single-zone convenience variant of `time_type/3` using the cached TZif terms; requires exactly one cached zone.

Compilation flags:
`static`

Template:
`time_type(UTC,TimeType)`
Mode and number of proofs:
`time_type(+types([integer,compound]),-compound)` - `one_or_error`

Exceptions:
No TZif terms are cached:
`existence_error(tzif_cache,tzif)`
The cached TZif terms do not contain exactly one zone:
`domain_error(single_zone_tzif_cache,TZifs)`

---

offset/4

Returns the UTC offset in seconds for a loaded zone and a UTC instant.

Compilation flags:
`static`

Template:
`offset(TZif,Zone,UTC,OffsetSeconds)`
Mode and number of proofs:
`offset(+compound,+atom,+types([integer,compound]),-integer)` - `zero_or_one`

---

offset/3

Returns the UTC offset in seconds for a zone in the cached TZif terms.

Compilation flags:
`static`

Template:
`offset(Zone,UTC,OffsetSeconds)`
Mode and number of proofs:
`offset(+atom,+types([integer,compound]),-integer)` - `one_or_error`

Exceptions:
`Zone` is not present in the cached TZif terms:
`existence_error(time_zone,Zone)`

---

offset/2

Cached single-zone convenience variant of `offset/3` using the cached TZif terms; requires exactly one cached zone.

Compilation flags:
`static`

Template:
`offset(UTC,OffsetSeconds)`
Mode and number of proofs:
`offset(+types([integer,compound]),-integer)` - `one_or_error`

Exceptions:
The cached TZif terms do not contain exactly one zone:
`domain_error(single_zone_tzif_cache,TZifs)`

---

daylight_saving_time/4

Returns `true` or `false` according to whether daylight saving time is active for a loaded zone and a UTC instant.

Compilation flags:
`static`

Template:
`daylight_saving_time(TZif,Zone,UTC,IsDST)`
Mode and number of proofs:
`daylight_saving_time(+compound,+atom,+types([integer,compound]),-atom)` - `zero_or_one`

---

daylight_saving_time/3

Returns daylight-saving information for a zone in the cached TZif terms.

Compilation flags:
`static`

Template:
`daylight_saving_time(Zone,UTC,IsDST)`
Mode and number of proofs:
`daylight_saving_time(+atom,+types([integer,compound]),-atom)` - `one_or_error`

Exceptions:
`Zone` is not present in the cached TZif terms:
`existence_error(time_zone,Zone)`

---

daylight_saving_time/2

Cached single-zone convenience variant of `daylight_saving_time/3` using the cached TZif terms; requires exactly one cached zone.

Compilation flags:
`static`

Template:
`daylight_saving_time(UTC,IsDST)`
Mode and number of proofs:
`daylight_saving_time(+types([integer,compound]),-atom)` - `one_or_error`

Exceptions:
The cached TZif terms do not contain exactly one zone:
`domain_error(single_zone_tzif_cache,TZifs)`

---

abbreviation/4

Returns the time-zone abbreviation for a loaded zone and a UTC instant.

Compilation flags:
`static`

Template:
`abbreviation(TZif,Zone,UTC,Abbreviation)`
Mode and number of proofs:
`abbreviation(+compound,+atom,+types([integer,compound]),-atom)` - `zero_or_one`

---

abbreviation/3

Returns the time-zone abbreviation for a zone in the cached TZif terms.

Compilation flags:
`static`

Template:
`abbreviation(Zone,UTC,Abbreviation)`
Mode and number of proofs:
`abbreviation(+atom,+types([integer,compound]),-atom)` - `one_or_error`

Exceptions:
`Zone` is not present in the cached TZif terms:
`existence_error(time_zone,Zone)`

---

abbreviation/2

Cached single-zone convenience variant of `abbreviation/3` using the cached TZif terms; requires exactly one cached zone.

Compilation flags:
`static`

Template:
`abbreviation(UTC,Abbreviation)`
Mode and number of proofs:
`abbreviation(+types([integer,compound]),-atom)` - `one_or_error`

Exceptions:
The cached TZif terms do not contain exactly one zone:
`domain_error(single_zone_tzif_cache,TZifs)`

---

local_time_type/4

Returns the applicable local time type for a loaded zone and a local civil time given as a `date_time/6` term. This strict variant fails unless the local civil time has a unique interpretation.

Compilation flags:
`static`

Template:
`local_time_type(TZif,Zone,LocalDateTime,TimeType)`
Mode and number of proofs:
`local_time_type(+compound,+atom,+compound,-compound)` - `zero_or_one`

---

local_time_type/3

Returns the applicable local time type for a zone in the cached TZif terms. This strict variant fails unless the local civil time has a unique interpretation.

Compilation flags:
`static`

Template:
`local_time_type(Zone,LocalDateTime,TimeType)`
Mode and number of proofs:
`local_time_type(+atom,+compound,-compound)` - `one_or_error`

Exceptions:
`LocalDateTime` is a variable:
`instantiation_error`
`LocalDateTime` is neither a variable nor a valid `date_time/6` term:
`type_error(date_time,LocalDateTime)`
`Zone` is not present in the cached TZif terms:
`existence_error(time_zone,Zone)`

---

local_time_type/2

Cached single-zone convenience variant of strict local civil-time lookup; requires exactly one cached zone and fails unless the local civil time has a unique interpretation.

Compilation flags:
`static`

Template:
`local_time_type(LocalDateTime,TimeType)`
Mode and number of proofs:

`local_time_type(+compound,-compound)` - `one_or_error`

Exceptions:

`LocalDateTime` is a variable:

`instantiation_error`

`LocalDateTime` is neither a variable nor a valid `date_time/6` term:

`type_error(date_time,LocalDateTime)`

No TZif terms are cached:

`existence_error(tzif_cache,tzif)`

The cached TZif terms do not contain exactly one zone:

`domain_error(single_zone_tzif_cache,TZifs)`

`local_time_type_with_resolution/5`

Returns the applicable local time type for a loaded zone and a local civil time using the explicit resolution mode: `strict` (fail unless exactly one interpretation exists), `first` (prefer the earliest valid interpretation), `second` (prefer the latest valid interpretation), and `all` (enumerate all valid interpretations in chronological order).

Compilation flags:

`static`

Template:

`local_time_type_with_resolution(TZif,Zone,LocalDateTime,ResolutionMode,TimeType)`

Mode and number of proofs:

`local_time_type_with_resolution(+compound,+atom,+compound,+atom,-compound)` - `zero_or_more`

`local_time_type_with_resolution/4`

Returns the applicable local time type for a zone in the cached TZif terms using the explicit resolution mode: `strict` (fail unless exactly one interpretation exists), `first` (prefer the earliest valid interpretation), `second` (prefer the latest valid interpretation), and `all` (enumerate all valid interpretations in chronological order).

Compilation flags:

`static`

Template:

`local_time_type_with_resolution(Zone,LocalDateTime,ResolutionMode,TimeType)`

Mode and number of proofs:

`local_time_type_with_resolution(+atom,+compound,+atom,-compound)` - `zero_or_more`

---

local_time_type_with_resolution/3

Cached single-zone convenience variant of local civil-time lookup using the explicit resolution mode: `strict` (fail unless exactly one interpretation exists), `first` (prefer the earliest valid interpretation), `second` (prefer the latest valid interpretation), and `all` (enumerate all valid interpretations in chronological order). Requires exactly one cached zone.

Compilation flags:

`static`

Template:

`local_time_type_with_resolution(LocalDateTime,ResolutionMode,TimeType)`

Mode and number of proofs:

`local_time_type_with_resolution(+compound,+atom,-compound)` - `zero_or_more`

---

local_offset/4

Returns the UTC offset in seconds for a loaded zone and a local civil time. This strict variant fails unless the local civil time has a unique interpretation.

Compilation flags:

`static`

Template:

`local_offset(TZif,Zone,LocalDateTime,OffsetSeconds)`

Mode and number of proofs:

`local_offset(+compound,+atom,+compound,-integer)` - `zero_or_one`

---

local_offset/3

Returns the UTC offset in seconds for a zone in the cached TZif terms. This strict variant fails unless the local civil time has a unique interpretation.

Compilation flags:

`static`

Template:

`local_offset(Zone,LocalDateTime,OffsetSeconds)`

Mode and number of proofs:

`local_offset(+atom,+compound,-integer)` - `one_or_error`

Exceptions:

`Zone` is not present in the cached TZif terms:

`existence_error(time_zone,Zone)`

---

local_offset/2

Cached single-zone convenience variant of strict local civil-time offset lookup.

Compilation flags:

```
static
```

Template:

```
local_offset(LocalDateTime,OffsetSeconds)
```

Mode and number of proofs:

```
local_offset(+compound,-integer) - one_or_error
```

Exceptions:

The cached TZif terms do not contain exactly one zone:

```
domain_error(single_zone_tzif_cache,TZifs)
```

---

local_offset_with_resolution/5

Returns the UTC offset in seconds for a loaded zone and a local civil time using the explicit resolution mode: `strict` (fail unless exactly one interpretation exists), `first` (prefer the earliest valid interpretation), `second` (prefer the latest valid interpretation), and `all` (enumerate all valid interpretations in chronological order).

Compilation flags:

```
static
```

Template:

```
local_offset_with_resolution(TZif,Zone,LocalDateTime,ResolutionMode,
OffsetSeconds)
```

Mode and number of proofs:

```
local_offset_with_resolution(+compound,+atom,+compound,+atom,-integer) -
zero_or_more
```

---

local_offset_with_resolution/4

Returns the UTC offset in seconds for a zone in the cached TZif terms using the explicit resolution mode: `strict` (fail unless exactly one interpretation exists), `first` (prefer the earliest valid interpretation), `second` (prefer the latest valid interpretation), and `all` (enumerate all valid interpretations in chronological order).

Compilation flags:

```
static
```

Template:

```
local_offset_with_resolution(Zone,LocalDateTime,ResolutionMode,OffsetSeconds)
```

Mode and number of proofs:

```
local_offset_with_resolution(+atom,+compound,+atom,-integer) - zero_or_more
```

---

local_offset_with_resolution/3

Cached single-zone convenience variant of local civil-time offset lookup using the explicit resolution mode; requires exactly one cached zone.

Compilation flags:
`static`

Template:
`local_offset_with_resolution(LocalDateTime,ResolutionMode,OffsetSeconds)`
Mode and number of proofs:
`local_offset_with_resolution(+compound,+atom,-integer)` - `zero_or_more`

---

local_daylight_saving_time/4

Returns daylight-saving information for a loaded zone and a local civil time. This strict variant fails unless the local civil time has a unique interpretation.

Compilation flags:
`static`

Template:
`local_daylight_saving_time(TZif,Zone,LocalDateTime,IsDST)`
Mode and number of proofs:
`local_daylight_saving_time(+compound,+atom,+compound,-atom)` - `zero_or_one`

---

local_daylight_saving_time/3

Returns daylight-saving information for a zone in the cached TZif terms. This strict variant fails unless the local civil time has a unique interpretation.

Compilation flags:
`static`

Template:
`local_daylight_saving_time(Zone,LocalDateTime,IsDST)`
Mode and number of proofs:
`local_daylight_saving_time(+atom,+compound,-atom)` - `one_or_error`

Exceptions:
`Zone` is not present in the cached TZif terms:
`existence_error(time_zone,Zone)`

---

local_daylight_saving_time/2

Cached single-zone convenience variant of strict local daylight-saving lookup.

Compilation flags:

```
static
```

Template:

```
local_daylight_saving_time(LocalDateTime,IsDST)
```

Mode and number of proofs:

```
local_daylight_saving_time(+compound,-atom) - one_or_error
```

Exceptions:

The cached TZif terms do not contain exactly one zone:

```
domain_error(single_zone_tzif_cache,TZifs)
```

---

local_daylight_saving_time_with_resolution/5

Returns daylight-saving information for a loaded zone and a local civil time using the explicit resolution mode: `strict` (fail unless exactly one interpretation exists), `first` (prefer the earliest valid interpretation), `second` (prefer the latest valid interpretation), and `all` (enumerate all valid interpretations in chronological order).

Compilation flags:

```
static
```

Template:

```
local_daylight_saving_time_with_resolution(TZif,Zone,LocalDateTime,
ResolutionMode,IsDST)
```

Mode and number of proofs:

```
local_daylight_saving_time_with_resolution(+compound,+atom,+compound,+atom,-atom)
- zero_or_more
```

---

local_daylight_saving_time_with_resolution/4

Returns daylight-saving information for a zone in the cached TZif terms using the explicit resolution mode: `strict` (fail unless exactly one interpretation exists), `first` (prefer the earliest valid interpretation), `second` (prefer the latest valid interpretation), and `all` (enumerate all valid interpretations in chronological order).

Compilation flags:

```
static
```

Template:

```
local_daylight_saving_time_with_resolution(Zone,LocalDateTime,ResolutionMode,
IsDST)
```

Mode and number of proofs:

```
local_daylight_saving_time_with_resolution(+atom,+compound,+atom,-atom) -
zero_or_more
```

---

local_daylight_saving_time_with_resolution/3

Cached single-zone convenience variant of local civil-time daylight-saving lookup using the explicit resolution mode; requires exactly one cached zone.

Compilation flags:

```
static
```

Template:

```
local_daylight_saving_time_with_resolution(LocalDateTime,ResolutionMode,IsDST)
```

Mode and number of proofs:

```
local_daylight_saving_time_with_resolution(+compound,+atom,-atom) - zero_or_more
```

---

local_abbreviation/4

Returns the time-zone abbreviation for a loaded zone and a local civil time. This strict variant fails unless the local civil time has a unique interpretation.

Compilation flags:

```
static
```

Template:

```
local_abbreviation(TZif,Zone,LocalDateTime,Abbreviation)
```

Mode and number of proofs:

```
local_abbreviation(+compound,+atom,+compound,-atom) - zero_or_one
```

---

local_abbreviation/3

Returns the time-zone abbreviation for a zone in the cached TZif terms. This strict variant fails unless the local civil time has a unique interpretation.

Compilation flags:

```
static
```

Template:

```
local_abbreviation(Zone,LocalDateTime,Abbreviation)
```

Mode and number of proofs:

```
local_abbreviation(+atom,+compound,-atom) - one_or_error
```

Exceptions:

`Zone` is not present in the cached TZif terms:
`existence_error(time_zone,Zone)`

---

local_abbreviation/2

Cached single-zone convenience variant of strict local abbreviation lookup.

Compilation flags:
`static`

Template:
`local_abbreviation(LocalDateTime,Abbreviation)`
Mode and number of proofs:
`local_abbreviation(+compound,-atom)` - `one_or_error`

Exceptions:
The cached TZif terms do not contain exactly one zone:
`domain_error(single_zone_tzif_cache,TZifs)`

---

local_abbreviation_with_resolution/5

Returns the time-zone abbreviation for a loaded zone and a local civil time using the explicit resolution mode: `strict` (fail unless exactly one interpretation exists), `first` (prefer the earliest valid interpretation), `second` (prefer the latest valid interpretation), and `all` (enumerate all valid interpretations in chronological order).

Compilation flags:
`static`

Template:
`local_abbreviation_with_resolution(TZif,Zone,LocalDateTime,ResolutionMode,Abbreviation)`
Mode and number of proofs:
`local_abbreviation_with_resolution(+compound,+atom,+compound,+atom,-atom)` - `zero_or_more`

---

local_abbreviation_with_resolution/4

Returns the time-zone abbreviation for a zone in the cached TZif terms using the explicit resolution mode: `strict` (fail unless exactly one interpretation exists), `first` (prefer the earliest valid interpretation), `second` (prefer the latest valid interpretation), and `all` (enumerate all valid interpretations in chronological order).

Compilation flags:
`static`

Template:
`local_abbreviation_with_resolution(Zone,LocalDateTime,ResolutionMode,Abbreviation)`
Mode and number of proofs:
`local_abbreviation_with_resolution(+atom,+compound,+atom,-atom)` - `zero_or_more`

---

local_abbreviation_with_resolution/3

Cached single-zone convenience variant of local civil-time abbreviation lookup using the explicit resolution mode; requires exactly one cached zone.

Compilation flags:
`static`

Template:
`local_abbreviation_with_resolution(LocalDateTime,ResolutionMode,Abbreviation)`
Mode and number of proofs:
`local_abbreviation_with_resolution(+compound,+atom,-atom)` - `zero_or_more`

---

local_time_type_reified/4

Returns a reified local civil-time lookup result for a loaded zone as one of `unique(TimeType)`, `ambiguous(TimeTypes)`, or `nonexistent`.

Compilation flags:
`static`

Template:
`local_time_type_reified(TZif,Zone,LocalDateTime,Result)`
Mode and number of proofs:
`local_time_type_reified(+compound,+atom,+compound,-compound)` - `one`

---

local_time_type_reified/3

Returns a reified local civil-time lookup result for a zone in the cached TZif terms as one of `unique(TimeType)`, `ambiguous(TimeTypes)`, or `nonexistent`.

Compilation flags:
`static`

Template:
`local_time_type_reified(Zone,LocalDateTime,Result)`
Mode and number of proofs:

`local_time_type_reified(+atom,+compound,-compound)` - `one_or_error`

Exceptions:

`LocalDateTime` is a variable:

`instantiation_error`

`LocalDateTime` is neither a variable nor a valid `date_time/6` term:

`type_error(date_time,LocalDateTime)`

`Zone` is not present in the cached TZif terms:

`existence_error(time_zone,Zone)`

`local_time_type_reified/2`

Cached single-zone convenience variant of reified local civil-time lookup; returns `unique(TimeType)`, `ambiguous(TimeTypes)`, or `nonexistent`.

Compilation flags:

`static`

Template:

`local_time_type_reified(LocalDateTime,Result)`

Mode and number of proofs:

`local_time_type_reified(+compound,-compound)` - `one_or_error`

Exceptions:

`LocalDateTime` is a variable:

`instantiation_error`

`LocalDateTime` is neither a variable nor a valid `date_time/6` term:

`type_error(date_time,LocalDateTime)`

No TZif terms are cached:

`existence_error(tzif_cache,tzif)`

The cached TZif terms do not contain exactly one zone:

`domain_error(single_zone_tzif_cache,TZifs)`

`local_offset_reified/4`

Returns a reified local offset lookup result for a loaded zone as one of `unique(OffsetSeconds)`, `ambiguous(OffsetSecondsList)`, or `nonexistent`.

Compilation flags:

`static`

Template:

`local_offset_reified(TZif,Zone,LocalDateTime,Result)`

Mode and number of proofs:

`local_offset_reified(+compound,+atom,+compound,-compound)` - `one`

---

local_offset_reified/3

Returns a reified local offset lookup result for a zone in the cached TZif terms as one of `unique(OffsetSeconds)`, `ambiguous(OffsetSecondsList)`, or `nonexistent`.

Compilation flags:
`static`

Template:
`local_offset_reified(Zone,LocalDateTime,Result)`
Mode and number of proofs:
`local_offset_reified(+atom,+compound,-compound)` - `one_or_error`

Exceptions:
`Zone` is not present in the cached TZif terms:
`existence_error(time_zone,Zone)`

---

local_offset_reified/2

Cached single-zone convenience variant of reified local offset lookup; returns `unique(OffsetSeconds)`, `ambiguous(OffsetSecondsList)`, or `nonexistent`.

Compilation flags:
`static`

Template:
`local_offset_reified(LocalDateTime,Result)`
Mode and number of proofs:
`local_offset_reified(+compound,-compound)` - `one_or_error`

Exceptions:
The cached TZif terms do not contain exactly one zone:
`domain_error(single_zone_tzif_cache,TZifs)`

---

local_daylight_saving_time_reified/4

Returns a reified local daylight-saving lookup result for a loaded zone as one of `unique(IsDST)`, `ambiguous(IsDSTList)`, or `nonexistent`.

Compilation flags:
`static`

Template:
`local_daylight_saving_time_reified(TZif,Zone,LocalDateTime,Result)`
Mode and number of proofs:

`local_daylight_saving_time_reified(+compound,+atom,+compound,-compound)` - `one`

`local_daylight_saving_time_reified/3`

Returns a reified local daylight-saving lookup result for a zone in the cached TZif terms as one of `unique(IsDST)`, `ambiguous(IsDSTList)`, or `nonexistent`.

Compilation flags:

`static`

Template:

`local_daylight_saving_time_reified(Zone,LocalDateTime,Result)`

Mode and number of proofs:

`local_daylight_saving_time_reified(+atom,+compound,-compound)` - `one_or_error`

Exceptions:

`Zone` is not present in the cached TZif terms:

`existence_error(time_zone,Zone)`

`local_daylight_saving_time_reified/2`

Cached single-zone convenience variant of reified local daylight-saving lookup; returns `unique(IsDST)`, `ambiguous(IsDSTList)`, or `nonexistent`.

Compilation flags:

`static`

Template:

`local_daylight_saving_time_reified(LocalDateTime,Result)`

Mode and number of proofs:

`local_daylight_saving_time_reified(+compound,-compound)` - `one_or_error`

Exceptions:

The cached TZif terms do not contain exactly one zone:

`domain_error(single_zone_tzif_cache,TZifs)`

---

local_abbreviation_reified/4

Returns a reified local abbreviation lookup result for a loaded zone as one of `unique(Abbreviation)`, `ambiguous(Abbreviations)`, or `nonexistent`.

Compilation flags:
`static`

Template:
`local_abbreviation_reified(TZif,Zone,LocalDateTime,Result)`
Mode and number of proofs:
`local_abbreviation_reified(+compound,+atom,+compound,-compound)` - `one`

---

local_abbreviation_reified/3

Returns a reified local abbreviation lookup result for a zone in the cached TZif terms as one of `unique(Abbreviation)`, `ambiguous(Abbreviations)`, or `nonexistent`.

Compilation flags:
`static`

Template:
`local_abbreviation_reified(Zone,LocalDateTime,Result)`
Mode and number of proofs:
`local_abbreviation_reified(+atom,+compound,-compound)` - `one_or_error`

Exceptions:
`Zone` is not present in the cached TZif terms:
`existence_error(time_zone,Zone)`

---

local_abbreviation_reified/2

Cached single-zone convenience variant of reified local abbreviation lookup; returns `unique(Abbreviation)`, `ambiguous(Abbreviations)`, or `nonexistent`.

Compilation flags:
`static`

Template:
`local_abbreviation_reified(LocalDateTime,Result)`
Mode and number of proofs:
`local_abbreviation_reified(+compound,-compound)` - `one_or_error`

Exceptions:
The cached TZif terms do not contain exactly one zone:

```
domain_error(single_zone_tzif_cache,TZifs)
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.340.3 `tzif_zone_ids`

Bundled canonical TZDB zone ids and backward-compatible aliases derived from IANA TZDB 2026a.

Availability:
`logtalk_load(tzif(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-04-07

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *tzdb_version/1*
  - *known_zone_id/1*
  - *zone_id_kind/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

tzdb_version/1

Bundled IANA TZDB release used to derive the zone-id table.

Compilation flags:
`static`

Template:
`tzdb_version(Version)`
Mode and number of proofs:
`tzdb_version(-atom)` - `one`

---

known_zone_id/1

True when the argument is a bundled canonical TZDB zone id or backward-compatible alias.

Compilation flags:
`static`

Template:
`known_zone_id(ZoneId)`
Mode and number of proofs:
`known_zone_id(+atom)` - `zero_or_one`

---

zone_id_kind/2

Classifies a bundled zone id as canonical or as a backward-compatible alias targeting another zone id.

Compilation flags:
`static`

Template:
`zone_id_kind(ZoneId,Kind)`
Mode and number of proofs:
`zone_id_kind(+atom,-compound)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.341 ulid

object

1.341.1 `ulid`

Universally Unique Lexicographically Sortable Identifier (ULID) generator using an atom representation.

Availability:
`logtalk_load(ulid(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-07-24

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *ulid(atom)*

Remarks:
(none)

Inherited public predicates:
generate/1 *generate/2* *generate/8* *timestamp/2* *timestamp/8*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ulid(Representation), *ulid_types*, *cuid2*, *ids*, *ksuid*, *nanoid*, *snowflakeid*, *typeid*, *uuid*

object

1.341.2 `ulid(Representation)`

- `Representation` - Text representation for the ULID. Possible values are `atom`, `chars`, and `codes`.

Universally Unique Lexicographically Sortable Identifier (ULID) generator.

Availability:
`logtalk_load(ulid(loader))`

Author: Paulo Moura
Version: 1:2:0
Date: 2026-07-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *ulid_protocol*
Uses:
crypto
date
os
type

Remarks:
(none)

Inherited public predicates:
generate/1 *generate/2* *generate/8* *timestamp/2* *timestamp/8*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ulid, *ulid_types*, *cuid2(Representation,Size,Alphabet)*, *ids(Representation,Bytes)*, *ksuid(Representation,Alphabet)*, *nanoid(Representation,Size,Alphabet)*, *snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceB*
typeid(Representation), *uuid(Representation)*

protocol

1.341.3 `ulid_protocol`

Universally Unique Lexicographically Sortable Identifier (ULID) generator protocol.

Availability:
`logtalk_load(ulid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-06

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:

(none)

- *Public predicates*
  - *generate/1*
  - *generate/2*
  - *generate/8*
  - *timestamp/2*
  - *timestamp/8*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

generate/1

Generates a new ULID.

Compilation flags:
`static`

Template:
`generate(ULID)`
Mode and number of proofs:
`generate(--ulid)` - `one_or_error`

Exceptions:
The representation parameter is a variable:
`instantiation_error`
The representation parameter is not a variable but also neither `atom`, `chars`, nor `codes`:
`domain_error(ulid_representation,Representation)`

---

generate/2

Generates a new ULID from a timestamp (number of milliseconds since the Unix epoch: 00:00:00 UTC on January 1, 1970).

Compilation flags:
`static`

Template:
`generate(Milliseconds,ULID)`
Mode and number of proofs:
`generate(+integer,--ulid)` - `one_or_error`

Exceptions:

The representation parameter is a variable:

```
instantiation_error
```

The representation parameter is not a variable but also neither `atom`, `chars`, nor `codes`:

```
domain_error(ulid_representation,Representation)
```

`Milliseconds` is a variable:

```
instantiation_error
```

`Milliseconds` is neither a variable nor an integer:

```
type_error(integer,Milliseconds)
```

`Milliseconds` is outside the 48-bit range required by the ULID specification:

```
domain_error(ulid_timestamp,Milliseconds)
```

---

generate/8

Generates a new ULID from a timestamp discrete components.

Compilation flags:

```
static
```

Template:

```
generate(Year,Month,Day,Hours,Minutes,Seconds,Milliseconds,ULID)
```

Mode and number of proofs:

```
generate(+integer,+integer,+integer,+integer,+integer,+integer,+integer,--ulid) -
one_or_error
```

Exceptions:

The representation parameter is a variable:

```
instantiation_error
```

The representation parameter is not a variable but also neither `atom`, `chars`, nor `codes`:

```
domain_error(ulid_representation,Representation)
```

`Milliseconds` is a variable:

```
instantiation_error
```

`Milliseconds` is neither a variable nor an integer:

```
type_error(integer,Milliseconds)
```

`Milliseconds` is an integer but outside the 0..999 range:

```
domain_error(ulid_millisecond,Milliseconds)
```

The timestamp is outside the 48-bit range required by the ULID specification:

```
domain_error(ulid_timestamp,Timestamp)
```

---

timestamp/2

Returns the given ULID timestamp (number of milliseconds since the Unix epoch: 00:00:00 UTC on January 1, 1970).

Compilation flags:
`static`

Template:
`timestamp(ULID,Milliseconds)`
Mode and number of proofs:
`timestamp(++ulid,-integer)` - `one_or_error`

Exceptions:
The representation parameter is a variable:
`instantiation_error`
The representation parameter is not a variable but also neither `atom`, `chars`, nor `codes`:
`domain_error(ulid_representation,Representation)`
`ULID` is a variable:
`instantiation_error`
`ULID` is neither a variable nor an atom when using the `atom` representation:
`type_error(atom,ULID)`
`ULID` is neither a variable nor a list of characters when using the `chars` representation:
`type_error(chars,ULID)`
`ULID` is neither a variable nor a list of character codes when using the `codes` representation:
`type_error(codes,ULID)`
`ULID` is not a valid ULID:
`domain_error(ulid,ULID)`

---

timestamp/8

Decodes a ULID into its timestamp discrete components.

Compilation flags:
`static`

Template:
`timestamp(ULID,Year,Month,Day,Hours,Minutes,Seconds,Milliseconds)`
Mode and number of proofs:
`timestamp(++ulid,-integer,-integer,-integer,-integer,-integer,-integer,-integer)` - `one_or_error`

Exceptions:
The representation parameter is a variable:
`instantiation_error`
The representation parameter is not a variable but also neither `atom`, `chars`, nor `codes`:
`domain_error(ulid_representation,Representation)`
`ULID` is a variable:

`instantiation_error`

`ULID` is neither a variable nor an atom when using the `atom` representation:

`type_error(atom,ULID)`

`ULID` is neither a variable nor a list of characters when using the `chars` representation:

`type_error(chars,ULID)`

`ULID` is neither a variable nor a list of character codes when using the `codes` representation:

`type_error(codes,ULID)`

`ULID` is not a valid ULID:

`domain_error(ulid,ULID)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

category

1.341.4 `ulid_types`

ULID type definition.

Availability:

`logtalk_load(ulid(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2023-05-19

Compilation flags:

`static`

Provides:

type::type/1

type::check/2

Uses:

list

type

Remarks:

- Provided types: This category adds a `ulid(Representation)` type for type-checking when using the `ulid` library object. Valid representation values are `atom`, `chars`, and `codes`.

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

ulid(Representation), *ulid*

1.342 umap_projection

object

1.342.1 `umap_projection`

Uniform Manifold Approximation and Projection dimension reducer for continuous, categorical, and mixed datasets.

Availability:
```
logtalk_load(umap_projection(loader))
```

Author: Paulo Moura
Version: 2:1:0
Date: 2026-10-24

Compilation flags:
```
static, context_switching_calls
```

Imports:
```
public dimension_reducer_common
```

Uses:

assignvars
avltree
binary_heap(Order)
fast_random(Algorithm)
format
linear_algebra
list
numberlist
pairs
population
type
user

Remarks:

(none)

Inherited public predicates:

check_dimension_reducer/1 *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *dimension_reducer_options/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *print_dimension_reducer/1* *transform/3* *valid_dimension_reducer/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
  - `graph_connected/2`
  - `normalized_adjacency/5`
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`graph_connected/2`

Checks fuzzy graph connectivity. Intended for use in testing.

Compilation flags:

`static`

Template:

`graph_connected(FuzzyGraph,SampleCount)`

Mode and number of proofs:

```
graph_connected(+list,+positive_integer) - zero_or_one
```

---

normalized_adjacency/5

Computes the normalized adjacency matrix for a starting vertex of zero and positive vertex degrees. Intended for use in testing.

Compilation flags:
```
static
```

Template:
```
normalized_adjacency(StartVertex,SampleCount,FuzzyGraph,Degrees,Matrix)
```
Mode and number of proofs:
```
normalized_adjacency(+integer,+positive_integer,+list,+list,-list) - zero_or_one
```

Operators

(none)

See also

tsne_projection, *spectral_clusterer*

1.343 union_find

object

1.343.1 union_find

Union find data structure implementation.

Availability:
```
logtalk_load(union_find(loader))
```

Author: José Antonio Riaza Valverde; adapted to Logtalk by Paulo Moura
Version: 1:0:0
Date: 2022-02-18

Compilation flags:
```
static, context_switching_calls
```

Implements:
```
public union_find_protocol
```
Extends:
```
public compound
```

Uses:
avltree

Remarks:
(none)

Inherited public predicates:
(<)/2 (=:=)/2 (=<)/2 (=\=)/2 (>)/2 (>=)/2 check/1 depth/2 disjoint_sets/2 find/4 find/5 ground/1 make_set/3 new/1 new/2 numbervars/1 numbervars/3 occurs/2 singletons/2 subsumes/2 subterm/2 union/4 union_all/3 valid/1 variables/2 variant/2 varnumbers/2 varnumbers/3

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.343.2 `union_find_protocol`

Union-find data structure protocol.

Availability:
`logtalk_load(union_find(loader))`

Author: José Antonio Riaza Valverde; adapted to Logtalk by Paulo Moura
Version: 1:0:0
Date: 2022-02-17

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *new/2*
  - *make_set/3*
  - *union/4*
  - *union_all/3*
  - *find/4*
  - *find/5*
  - *disjoint_sets/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

new/2

Creates a new union-find data structure with a list of elements as keys.

Compilation flags:
`static`

Template:
`new(Elements,UnionFind)`
Mode and number of proofs:
`new(+list(element),?union_find)` - `zero_or_one`

---

make_set/3

Makes a new set by creating a new element with a unique key `Element`, a rank of `0`, and a parent pointer to itself. The parent pointer to itself indicates that the element is the representative member of its own set.

Compilation flags:
`static`

Template:
`make_set(UnionFind,Element,NewUnionFind)`
Mode and number of proofs:
`make_set(+union_find,+element,?union_find)` - `zero_or_one`

---

union/4

Merges the two trees, if distinct, that contain the given elements. The trees are joined by attaching the shorter tree (by rank) to the root of the taller tree. Fails if any of the elements is not found.

Compilation flags:

```
static
```

Template:

```
union(UnionFind,Element1,Element2,NewUnionFind)
```

Mode and number of proofs:

```
union(+union_find,+element,+element,?union_find) - zero_or_one
```

---

union_all/3

Merges the distinct trees for all the given elements returning the resulting union-find data structure. Fails if any of the elements is not found.

Compilation flags:

```
static
```

Template:

```
union_all(UnionFind,Elements,NewUnionFind)
```

Mode and number of proofs:

```
union_all(+union_find,+list(element),?union_find) - zero_or_one
```

---

find/4

Finds the root element of a set by following the chain of parent pointers from the given element. Root is the representative member of the set to which the element belongs, and may be element itself. Fails if the element is not found.

Compilation flags:

```
static
```

Template:

```
find(UnionFind,Element,Root,NewUnionFind)
```

Mode and number of proofs:

```
find(+union_find,+element,?element,?union_find) - zero_or_one
```

Remarks:

- Path compression: The structure of the tree containing the element is flattened by making every node point to the root whenever this predicate is used on it.

---

find/5

Same as the `find/4` predicate, but returning also the rank of the root. Fails if the element is not found.

Compilation flags:
`static`

Template:
`find(UnionFind,Element,Root,Rank,UnionFindOut)`
Mode and number of proofs:
`find(+union_find,+element,?element,?rank,?union_find)` - `zero_or_one`

Remarks:

- Path compression: The structure of the tree containing the element is flattened by making every node point to the root whenever this predicate is used on it.

---

disjoint_sets/2

Returns the list of disjoint sets in the given union-find data structure.

Compilation flags:
`static`

Template:
`disjoint_sets(UnionFind,Sets)`
Mode and number of proofs:
`disjoint_sets(+union_find,?sets)` - `zero_or_one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

union_find

1.344 univariate_distributions

object

1.344.1 `univariate_distributions(Random)`

- `Random` - An object implementing the `sampling_protocol` protocol.

Univariate probability distributions using an injected random source.

Availability:
`logtalk_load(univariate_distributions(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-12

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *univariate_distributions_protocol*
Uses:
type

Remarks:

- Normal cumulative distribution: Uses the Abramowitz and Stegun formula 26.2.17 approximation.
- Normal quantile: Uses the Peter J. Acklam rational approximation.

Inherited public predicates:
beta/3 beta_density/4 beta_distribution/4 beta_log_density/4 beta_quantile/4 beta_samples/4 chi_squared/2 chi_squared_density/3 chi_squared_distribution/3 chi_squared_log_density/3 chi_squared_quantile/3 chi_squared_samples/3 exponential/2 exponential_density/3 exponential_distribution/3 exponential_log_density/3 exponential_quantile/3 exponential_samples/3 fisher/3 fisher_density/4 fisher_distribution/4 fisher_log_density/4 fisher_quantile/4 fisher_samples/4 gamma/3 gamma_density/4 gamma_distribution/4 gamma_log_density/4 gamma_quantile/4 gamma_samples/4 normal/3 normal_density/4 normal_distribution/4 normal_log_density/4 normal_quantile/4 normal_samples/4 standard_normal/1 standard_normal_density/2 standard_normal_distribution/2 standard_normal_log_density/2 standard_normal_quantile/2 standard_normal_samples/2 standard_t/2 standard_t_density/3 standard_t_distribution/3 standard_t_log_density/3 standard_t_quantile/3 standard_t_samples/3 t/4 t_density/5 t_distribution/5 t_log_density/5 t_quantile/5 t_samples/5

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

univariate_distributions_protocol, *multivariate_distributions(Random)*, *random(Algorithm)*, *fast_random(Algorithm)*, *fast_random*, *backend_random*

protocol

1.344.2 `univariate_distributions_protocol`

Univariate probability distribution predicates.

Availability:
`logtalk_load(univariate_distributions(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-08-12

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *standard_normal/1*
  - *standard_normal_samples/2*
  - *normal/3*

```
      - normal_samples/4
      - standard_normal_density/2
      - standard_normal_log_density/2
      - standard_normal_distribution/2
      - standard_normal_quantile/2
      - normal_density/4
      - normal_log_density/4
      - normal_distribution/4
      - normal_quantile/4
      - standard_t/2
      - standard_t_samples/3
      - standard_t_density/3
      - standard_t_log_density/3
      - standard_t_distribution/3
      - standard_t_quantile/3
      - t/4
      - t_samples/5
      - t_density/5
      - t_log_density/5
      - t_distribution/5
      - t_quantile/5
      - chi_squared/2
      - chi_squared_samples/3
      - chi_squared_density/3
      - chi_squared_log_density/3
      - chi_squared_distribution/3
      - chi_squared_quantile/3
      - gamma/3
      - gamma_samples/4
      - gamma_density/4
      - gamma_log_density/4
      - gamma_distribution/4
      - gamma_quantile/4
      - beta/3
      - beta_samples/4
      - beta_density/4
      - beta_log_density/4
      - beta_distribution/4
      - beta_quantile/4
      - exponential/2
      - exponential_samples/3
      - exponential_density/3
      - exponential_log_density/3
      - exponential_distribution/3
      - exponential_quantile/3
      - fisher/3
      - fisher_samples/4
```

    - *fisher_density/4*
    - *fisher_log_density/4*
    - *fisher_distribution/4*
    - *fisher_quantile/4*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

standard_normal/1

Returns a standard normally distributed random value.

Compilation flags:
`static`

Template:
`standard_normal(Value)`
Mode and number of proofs:
`standard_normal(-float)` - `one`

---

standard_normal_samples/2

Returns the requested number of standard normally distributed random values.

Compilation flags:
`static`

Template:
`standard_normal_samples(Count,Samples)`
Mode and number of proofs:
`standard_normal_samples(+integer,-list(float))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Count)`

---

normal/3

Returns a normally distributed random value with the given mean and standard deviation. A zero deviation returns the mean.

Compilation flags:

```
static
```

Template:

```
normal(Mean,Deviation,Value)
```

Mode and number of proofs:

```
normal(+float,+non_negative_float,-float) - one_or_error
```

Exceptions:

`Mean` is a variable:

```
instantiation_error
```

`Mean` is neither a variable nor a float:

```
type_error(float,Mean)
```

`Deviation` is a variable:

```
instantiation_error
```

`Deviation` is neither a variable nor a float:

```
type_error(float,Deviation)
```

`Deviation` is a negative float:

```
domain_error(non_negative_float,Deviation)
```

---

normal_samples/4

Returns the requested number of normally distributed random values with the given mean and standard deviation.

Compilation flags:

```
static
```

Template:

```
normal_samples(Count,Mean,Deviation,Samples)
```

Mode and number of proofs:

```
normal_samples(+integer,+float,+non_negative_float,-list(float)) - one_or_error
```

Exceptions:

`Count` is a variable:

```
instantiation_error
```

`Count` is neither a variable nor an integer:

```
type_error(integer,Count)
```

`Count` is an integer but not a non-negative integer:

```
domain_error(non_negative_integer,Count)
```

`Mean` is a variable:

```
instantiation_error
```

`Mean` is neither a variable nor a float:

```
type_error(float,Mean)
```

`Deviation` is a variable:
`instantiation_error`
`Deviation` is neither a variable nor a float:
`type_error(float,Deviation)`
`Deviation` is a negative float:
`domain_error(non_negative_float,Deviation)`

---

standard_normal_density/2

Computes the standard normal probability density at the given value.

Compilation flags:
`static`

Template:
`standard_normal_density(Value,Density)`
Mode and number of proofs:
`standard_normal_density(+number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`

---

standard_normal_log_density/2

Computes the standard normal log-density at the given value.

Compilation flags:
`static`

Template:
`standard_normal_log_density(Value,LogDensity)`
Mode and number of proofs:
`standard_normal_log_density(+number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`

---

standard_normal_distribution/2

Computes an approximation of the standard normal cumulative distribution at the given value.

Compilation flags:
`static`

Template:
`standard_normal_distribution(Value,Probability)`
Mode and number of proofs:
`standard_normal_distribution(+number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`

---

standard_normal_quantile/2

Computes an approximation of the standard normal quantile for a probability strictly between zero and one.

Compilation flags:
`static`

Template:
`standard_normal_quantile(Probability,Quantile)`
Mode and number of proofs:
`standard_normal_quantile(+open_probability,-float)` - `one_or_error`

Exceptions:
`Probability` is a variable:
`instantiation_error`
`Probability` is neither a variable nor a float:
`type_error(float,Probability)`
`Probability` is a float but not strictly between zero and one:
`domain_error(open_probability,Probability)`

---

normal_density/4

Computes the normal probability density at a value for the given mean and positive standard deviation.

Compilation flags:
`static`

Template:
`normal_density(Value,Mean,Deviation,Density)`
Mode and number of proofs:
`normal_density(+number,+number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Mean` is a variable:
`instantiation_error`
`Mean` is neither a variable nor a number:
`type_error(number,Mean)`
`Deviation` is a variable:
`instantiation_error`
`Deviation` is neither a variable nor a number:
`type_error(number,Deviation)`
`Deviation` is a number but not a positive number:
`domain_error(positive_number,Deviation)`

---

normal_log_density/4

Computes the normal log-density at a value for the given mean and positive standard deviation.

Compilation flags:
`static`

Template:
`normal_log_density(Value,Mean,Deviation,LogDensity)`
Mode and number of proofs:
`normal_log_density(+number,+number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Mean` is a variable:
`instantiation_error`

`Mean` is neither a variable nor a number:
`type_error(number,Mean)`
`Deviation` is a variable:
`instantiation_error`
`Deviation` is neither a variable nor a number:
`type_error(number,Deviation)`
`Deviation` is a number but not a positive number:
`domain_error(positive_number,Deviation)`

`normal_distribution/4`

Computes an approximation of the normal cumulative distribution at a value for the given mean and positive standard deviation.

Compilation flags:
`static`

Template:
`normal_distribution(Value,Mean,Deviation,Probability)`
Mode and number of proofs:
`normal_distribution(+number,+number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Mean` is a variable:
`instantiation_error`
`Mean` is neither a variable nor a number:
`type_error(number,Mean)`
`Deviation` is a variable:
`instantiation_error`
`Deviation` is neither a variable nor a number:
`type_error(number,Deviation)`
`Deviation` is a number but not a positive number:
`domain_error(positive_number,Deviation)`

---

normal_quantile/4

Computes an approximation of the normal quantile for a probability strictly between zero and one and the given mean and positive standard deviation.

Compilation flags:
`static`

Template:
`normal_quantile(Probability,Mean,Deviation,Quantile)`
Mode and number of proofs:
`normal_quantile(+open_probability,+number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Probability` is a variable:
`instantiation_error`
`Probability` is neither a variable nor a float:
`type_error(float,Probability)`
`Probability` is a float but not strictly between zero and one:
`domain_error(open_probability,Probability)`
`Mean` is a variable:
`instantiation_error`
`Mean` is neither a variable nor a number:
`type_error(number,Mean)`
`Deviation` is a variable:
`instantiation_error`
`Deviation` is neither a variable nor a number:
`type_error(number,Deviation)`
`Deviation` is a number but not a positive number:
`domain_error(positive_number,Deviation)`

---

standard_t/2

Returns a standard Student's t distributed random value.

Compilation flags:
`static`

Template:
`standard_t(DegreesOfFreedom,Value)`
Mode and number of proofs:
`standard_t(+positive_number,-float)` - `one_or_error`

Exceptions:
`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:

`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`

---

standard_t_samples/3

Returns the requested number of standard Student's t distributed random values.

Compilation flags:
`static`

Template:
`standard_t_samples(Count,DegreesOfFreedom,Samples)`
Mode and number of proofs:
`standard_t_samples(+integer,+positive_number,-list(float))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Count)`
`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`

---

standard_t_density/3

Computes the standard Student's t probability density at the given value.

Compilation flags:
`static`

Template:
`standard_t_density(Value,DegreesOfFreedom,Density)`
Mode and number of proofs:
`standard_t_density(+number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:

instantiation_error
Value is neither a variable nor a number:
type_error(number,Value)
DegreesOfFreedom is a variable:
instantiation_error
DegreesOfFreedom is neither a variable nor a number:
type_error(number,DegreesOfFreedom)
DegreesOfFreedom is a number but not a positive number:
domain_error(positive_number,DegreesOfFreedom)

---

standard_t_log_density/3

Computes the standard Student's t log-density at the given value.

Compilation flags:
static

Template:
standard_t_log_density(Value,DegreesOfFreedom,LogDensity)
Mode and number of proofs:
standard_t_log_density(+number,+positive_number,-float) - one_or_error

Exceptions:
Value is a variable:
instantiation_error
Value is neither a variable nor a number:
type_error(number,Value)
DegreesOfFreedom is a variable:
instantiation_error
DegreesOfFreedom is neither a variable nor a number:
type_error(number,DegreesOfFreedom)
DegreesOfFreedom is a number but not a positive number:
domain_error(positive_number,DegreesOfFreedom)

---

standard_t_distribution/3

Computes an approximation of the standard Student's t cumulative distribution at the given value.

Compilation flags:
static

Template:
standard_t_distribution(Value,DegreesOfFreedom,Probability)
Mode and number of proofs:
standard_t_distribution(+number,+positive_number,-float) - one_or_error

Exceptions:

`Value` is a variable:

`instantiation_error`

`Value` is neither a variable nor a number:

`type_error(number,Value)`

`DegreesOfFreedom` is a variable:

`instantiation_error`

`DegreesOfFreedom` is neither a variable nor a number:

`type_error(number,DegreesOfFreedom)`

`DegreesOfFreedom` is a number but not a positive number:

`domain_error(positive_number,DegreesOfFreedom)`

`standard_t_quantile/3`

Computes an approximation of the standard Student's t quantile for a probability strictly between zero and one.

Compilation flags:

`static`

Template:

`standard_t_quantile(Probability,DegreesOfFreedom,Quantile)`

Mode and number of proofs:

`standard_t_quantile(+open_probability,+positive_number,-float)` - `one_or_error`

Exceptions:

`Probability` is a variable:

`instantiation_error`

`Probability` is neither a variable nor a float:

`type_error(float,Probability)`

`Probability` is a float but not strictly between zero and one:

`domain_error(open_probability,Probability)`

`DegreesOfFreedom` is a variable:

`instantiation_error`

`DegreesOfFreedom` is neither a variable nor a number:

`type_error(number,DegreesOfFreedom)`

`DegreesOfFreedom` is a number but not a positive number:

`domain_error(positive_number,DegreesOfFreedom)`

---

t/4

Returns a location-scale Student's t distributed random value.

Compilation flags:
`static`

Template:
`t(Location,Scale,DegreesOfFreedom,Value)`
Mode and number of proofs:
`t(+number,+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Location` is a variable:
`instantiation_error`
`Location` is neither a variable nor a number:
`type_error(number,Location)`
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`
`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`

---

t_samples/5

Returns the requested number of location-scale Student's t distributed random values.

Compilation flags:
`static`

Template:
`t_samples(Count,Location,Scale,DegreesOfFreedom,Samples)`
Mode and number of proofs:
`t_samples(+integer,+number,+positive_number,+positive_number,-list(float))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:

`type_error(integer,Count)`

`Count` is an integer but not a non-negative integer:

`domain_error(non_negative_integer,Count)`

`Location` is a variable:

`instantiation_error`

`Location` is neither a variable nor a number:

`type_error(number,Location)`

`Scale` is a variable:

`instantiation_error`

`Scale` is neither a variable nor a number:

`type_error(number,Scale)`

`Scale` is not positive:

`domain_error(positive_number,Scale)`

`DegreesOfFreedom` is a variable:

`instantiation_error`

`DegreesOfFreedom` is neither a variable nor a number:

`type_error(number,DegreesOfFreedom)`

`DegreesOfFreedom` is a number but not a positive number:

`domain_error(positive_number,DegreesOfFreedom)`

`t_density/5`

Computes the location-scale Student's t probability density.

Compilation flags:

`static`

Template:

`t_density(Value,Location,Scale,DegreesOfFreedom,Density)`

Mode and number of proofs:

`t_density(+number,+number,+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:

`Value` is a variable:

`instantiation_error`

`Value` is neither a variable nor a number:

`type_error(number,Value)`

`Location` is a variable:

`instantiation_error`

`Location` is neither a variable nor a number:

`type_error(number,Location)`

`Scale` is a variable:

`instantiation_error`

`Scale` is neither a variable nor a number:

`type_error(number,Scale)`

`Scale` is not positive:

`domain_error(positive_number,Scale)`

`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`

---

t_log_density/5

Computes the location-scale Student's t log-density.

Compilation flags:
`static`

Template:
`t_log_density(Value,Location,Scale,DegreesOfFreedom,LogDensity)`
Mode and number of proofs:
`t_log_density(+number,+number,+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Location` is a variable:
`instantiation_error`
`Location` is neither a variable nor a number:
`type_error(number,Location)`
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`
`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`

---

t_distribution/5

Computes an approximation of the location-scale Student's t cumulative distribution.

Compilation flags:
`static`

Template:
`t_distribution(Value,Location,Scale,DegreesOfFreedom,Probability)`
Mode and number of proofs:
`t_distribution(+number,+number,+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Location` is a variable:
`instantiation_error`
`Location` is neither a variable nor a number:
`type_error(number,Location)`
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`
`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`

---

t_quantile/5

Computes an approximation of the location-scale Student's t quantile for a probability strictly between zero and one.

Compilation flags:
`static`

Template:
`t_quantile(Probability,Location,Scale,DegreesOfFreedom,Quantile)`
Mode and number of proofs:
`t_quantile(+open_probability,+number,+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:

`Probability` is a variable:
`instantiation_error`
`Probability` is neither a variable nor a float:
`type_error(float,Probability)`
`Probability` is a float but not strictly between zero and one:
`domain_error(open_probability,Probability)`
`Location` is a variable:
`instantiation_error`
`Location` is neither a variable nor a number:
`type_error(number,Location)`
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`
`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`

`chi_squared/2`

Returns a chi-squared distributed random value.

Compilation flags:

`static`

Template:

`chi_squared(DegreesOfFreedom,Value)`

Mode and number of proofs:

`chi_squared(+positive_number,-float)` - `one_or_error`

Exceptions:

`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`

---

chi_squared_samples/3

Returns the requested number of chi-squared distributed random values.

Compilation flags:
`static`

Template:
`chi_squared_samples(Count,DegreesOfFreedom,Samples)`
Mode and number of proofs:
`chi_squared_samples(+integer,+positive_number,-list(float))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Count)`
`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom)`
`DegreesOfFreedom` is a number but not a positive number:
`domain_error(positive_number,DegreesOfFreedom)`

---

chi_squared_density/3

Computes the chi-squared probability density at the given value. At zero, returns the atom `positive_infinity` when the degrees of freedom are smaller than two.

Compilation flags:
`static`

Template:
`chi_squared_density(Value,DegreesOfFreedom,Density)`
Mode and number of proofs:
`chi_squared_density(+number,+positive_number,-atomic)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`DegreesOfFreedom` is a variable:
`instantiation_error`
`DegreesOfFreedom` is neither a variable nor a number:

type_error(number,DegreesOfFreedom)
DegreesOfFreedom is a number but not a positive number:
domain_error(positive_number,DegreesOfFreedom)

---

chi_squared_log_density/3

Computes the chi-squared log-density at the given value. Returns the atoms positive_infinity or negative_infinity when the density is respectively infinite or zero.

Compilation flags:
static

Template:
chi_squared_log_density(Value,DegreesOfFreedom,LogDensity)
Mode and number of proofs:
chi_squared_log_density(+number,+positive_number,-atomic) - one_or_error

Exceptions:
Value is a variable:
instantiation_error
Value is neither a variable nor a number:
type_error(number,Value)
DegreesOfFreedom is a variable:
instantiation_error
DegreesOfFreedom is neither a variable nor a number:
type_error(number,DegreesOfFreedom)
DegreesOfFreedom is a number but not a positive number:
domain_error(positive_number,DegreesOfFreedom)

---

chi_squared_distribution/3

Computes an approximation of the chi-squared cumulative distribution at the given value.

Compilation flags:
static

Template:
chi_squared_distribution(Value,DegreesOfFreedom,Probability)
Mode and number of proofs:
chi_squared_distribution(+number,+positive_number,-float) - one_or_error

Exceptions:
Value is a variable:
instantiation_error
Value is neither a variable nor a number:

type_error(number,Value)
DegreesOfFreedom is a variable:
instantiation_error
DegreesOfFreedom is neither a variable nor a number:
type_error(number,DegreesOfFreedom)
DegreesOfFreedom is a number but not a positive number:
domain_error(positive_number,DegreesOfFreedom)

---

chi_squared_quantile/3

Computes an approximation of the chi-squared quantile for a probability strictly between zero and one.

Compilation flags:
static

Template:
chi_squared_quantile(Probability,DegreesOfFreedom,Quantile)
Mode and number of proofs:
chi_squared_quantile(+open_probability,+positive_number,-float) - one_or_error

Exceptions:
Probability is a variable:
instantiation_error
Probability is neither a variable nor a float:
type_error(float,Probability)
Probability is a float but not strictly between zero and one:
domain_error(open_probability,Probability)
DegreesOfFreedom is a variable:
instantiation_error
DegreesOfFreedom is neither a variable nor a number:
type_error(number,DegreesOfFreedom)
DegreesOfFreedom is a number but not a positive number:
domain_error(positive_number,DegreesOfFreedom)

---

gamma/3

Returns a gamma distributed random value (shape-scale parameterization).

Compilation flags:
static

Template:
gamma(Shape,Scale,Value)
Mode and number of proofs:
gamma(+positive_number,+positive_number,-float) - one_or_error

Exceptions:

`Shape` is a variable:

`instantiation_error`

`Shape` is neither a variable nor a number:

`type_error(number,Shape)`

`Shape` is not positive:

`domain_error(positive_number,Shape)`

`Scale` is a variable:

`instantiation_error`

`Scale` is neither a variable nor a number:

`type_error(number,Scale)`

`Scale` is not positive:

`domain_error(positive_number,Scale)`

---

gamma_samples/4

Returns the requested number of gamma distributed random values.

Compilation flags:

`static`

Template:

`gamma_samples(Count,Shape,Scale,Samples)`

Mode and number of proofs:

`gamma_samples(+integer,+positive_number,+positive_number,-list(float))` - `one_or_error`

Exceptions:

`Count` is a variable:

`instantiation_error`

`Count` is neither a variable nor an integer:

`type_error(integer,Count)`

`Count` is an integer but not a non-negative integer:

`domain_error(non_negative_integer,Count)`

`Shape` is a variable:

`instantiation_error`

`Shape` is neither a variable nor a number:

`type_error(number,Shape)`

`Shape` is not positive:

`domain_error(positive_number,Shape)`

`Scale` is a variable:

`instantiation_error`

`Scale` is neither a variable nor a number:

`type_error(number,Scale)`

`Scale` is not positive:

`domain_error(positive_number,Scale)`

---

gamma_density/4

Computes the gamma probability density at the given value. At zero, returns the atom positive_infinity when the shape is smaller than one.

Compilation flags:

static

Template:

gamma_density(Value,Shape,Scale,Density)

Mode and number of proofs:

gamma_density(+number,+positive_number,+positive_number,-atomic) - one_or_error

Exceptions:

Value is a variable:

instantiation_error

Value is neither a variable nor a number:

type_error(number,Value)

Shape is a variable:

instantiation_error

Shape is neither a variable nor a number:

type_error(number,Shape)

Shape is not positive:

domain_error(positive_number,Shape)

Scale is a variable:

instantiation_error

Scale is neither a variable nor a number:

type_error(number,Scale)

Scale is not positive:

domain_error(positive_number,Scale)

---

gamma_log_density/4

Computes the gamma log-density at the given value. Returns the atoms positive_infinity or negative_infinity when the density is respectively infinite or zero.

Compilation flags:

static

Template:

gamma_log_density(Value,Shape,Scale,LogDensity)

Mode and number of proofs:

gamma_log_density(+number,+positive_number,+positive_number,-atomic) - one_or_error

Exceptions:

Value is a variable:

instantiation_error

`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Shape` is a variable:
`instantiation_error`
`Shape` is neither a variable nor a number:
`type_error(number,Shape)`
`Shape` is not positive:
`domain_error(positive_number,Shape)`
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`

---

gamma_distribution/4

Computes an approximation of the gamma cumulative distribution at the given value.

Compilation flags:
`static`

Template:
`gamma_distribution(Value,Shape,Scale,Probability)`
Mode and number of proofs:
`gamma_distribution(+number,+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Shape` is a variable:
`instantiation_error`
`Shape` is neither a variable nor a number:
`type_error(number,Shape)`
`Shape` is not positive:
`domain_error(positive_number,Shape)`
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`

---

gamma_quantile/4

Computes an approximation of the gamma quantile for a probability strictly between zero and one.

Compilation flags:
`static`

Template:
`gamma_quantile(Probability,Shape,Scale,Quantile)`
Mode and number of proofs:
`gamma_quantile(+open_probability,+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Probability` is a variable:
`instantiation_error`
`Probability` is neither a variable nor a float:
`type_error(float,Probability)`
`Probability` is a float but not strictly between zero and one:
`domain_error(open_probability,Probability)`
`Shape` is a variable:
`instantiation_error`
`Shape` is neither a variable nor a number:
`type_error(number,Shape)`
`Shape` is not positive:
`domain_error(positive_number,Shape)`
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`

---

beta/3

Returns a beta distributed random value.

Compilation flags:
`static`

Template:
`beta(Alpha,Beta,Value)`
Mode and number of proofs:
`beta(+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Alpha` is a variable:

`instantiation_error`
`Alpha` is neither a variable nor a number:
`type_error(number,Alpha)`
`Alpha` is not positive:
`domain_error(positive_number,Alpha)`
`Beta` is a variable:
`instantiation_error`
`Beta` is neither a variable nor a number:
`type_error(number,Beta)`
`Beta` is not positive:
`domain_error(positive_number,Beta)`

---

beta_samples/4

Returns the requested number of beta distributed random values.

Compilation flags:
`static`

Template:
`beta_samples(Count,Alpha,Beta,Samples)`
Mode and number of proofs:
`beta_samples(+integer,+positive_number,+positive_number,-list(float))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Count)`
`Alpha` is a variable:
`instantiation_error`
`Alpha` is neither a variable nor a number:
`type_error(number,Alpha)`
`Alpha` is not positive:
`domain_error(positive_number,Alpha)`
`Beta` is a variable:
`instantiation_error`
`Beta` is neither a variable nor a number:
`type_error(number,Beta)`
`Beta` is not positive:
`domain_error(positive_number,Beta)`

---

beta_density/4

Computes the beta probability density at the given value. At either endpoint, returns the atom positive_infinity when the corresponding shape parameter is smaller than one.

Compilation flags:

static

Template:

beta_density(Value,Alpha,Beta,Density)

Mode and number of proofs:

beta_density(+number,+positive_number,+positive_number,-atomic) - one_or_error

Exceptions:

Value is a variable:

instantiation_error

Value is neither a variable nor a number:

type_error(number,Value)

Alpha is a variable:

instantiation_error

Alpha is neither a variable nor a number:

type_error(number,Alpha)

Alpha is not positive:

domain_error(positive_number,Alpha)

Beta is a variable:

instantiation_error

Beta is neither a variable nor a number:

type_error(number,Beta)

Beta is not positive:

domain_error(positive_number,Beta)

---

beta_log_density/4

Computes the beta log-density at the given value. Returns the atoms positive_infinity or negative_infinity when the density is respectively infinite or zero.

Compilation flags:

static

Template:

beta_log_density(Value,Alpha,Beta,LogDensity)

Mode and number of proofs:

beta_log_density(+number,+positive_number,+positive_number,-atomic) - one_or_error

Exceptions:

Value is a variable:

instantiation_error

`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Alpha` is a variable:
`instantiation_error`
`Alpha` is neither a variable nor a number:
`type_error(number,Alpha)`
`Alpha` is not positive:
`domain_error(positive_number,Alpha)`
`Beta` is a variable:
`instantiation_error`
`Beta` is neither a variable nor a number:
`type_error(number,Beta)`
`Beta` is not positive:
`domain_error(positive_number,Beta)`

---

beta_distribution/4

Computes an approximation of the beta cumulative distribution at the given value.

Compilation flags:
`static`

Template:
`beta_distribution(Value,Alpha,Beta,Probability)`
Mode and number of proofs:
`beta_distribution(+number,+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Alpha` is a variable:
`instantiation_error`
`Alpha` is neither a variable nor a number:
`type_error(number,Alpha)`
`Alpha` is not positive:
`domain_error(positive_number,Alpha)`
`Beta` is a variable:
`instantiation_error`
`Beta` is neither a variable nor a number:
`type_error(number,Beta)`
`Beta` is not positive:
`domain_error(positive_number,Beta)`

---

beta_quantile/4

Computes an approximation of the beta quantile for a probability strictly between zero and one.

Compilation flags:
`static`

Template:
`beta_quantile(Probability,Alpha,Beta,Quantile)`
Mode and number of proofs:
`beta_quantile(+open_probability,+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Probability` is a variable:
`instantiation_error`
`Probability` is neither a variable nor a float:
`type_error(float,Probability)`
`Probability` is a float but not strictly between zero and one:
`domain_error(open_probability,Probability)`
`Alpha` is a variable:
`instantiation_error`
`Alpha` is neither a variable nor a number:
`type_error(number,Alpha)`
`Alpha` is not positive:
`domain_error(positive_number,Alpha)`
`Beta` is a variable:
`instantiation_error`
`Beta` is neither a variable nor a number:
`type_error(number,Beta)`
`Beta` is not positive:
`domain_error(positive_number,Beta)`

---

exponential/2

Returns an exponentially distributed random value (scale parameterization).

Compilation flags:
`static`

Template:
`exponential(Scale,Value)`
Mode and number of proofs:
`exponential(+positive_number,-float)` - `one_or_error`

Exceptions:
`Scale` is a variable:

`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`

---

exponential_samples/3

Returns the requested number of exponentially distributed random values.

Compilation flags:
`static`

Template:
`exponential_samples(Count,Scale,Samples)`
Mode and number of proofs:
`exponential_samples(+integer,+positive_number,-list(float))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Count)`
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`

---

exponential_density/3

Computes the exponential probability density at the given value.

Compilation flags:
`static`

Template:
`exponential_density(Value,Scale,Density)`
Mode and number of proofs:
`exponential_density(+number,+positive_number,-float)` - `one_or_error`

Exceptions:

`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`

---

exponential_log_density/3

Computes the exponential log-density at the given value.

Compilation flags:
`static`

Template:
`exponential_log_density(Value,Scale,LogDensity)`
Mode and number of proofs:
`exponential_log_density(+number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Value` is a variable:
`instantiation_error`
`Value` is neither a variable nor a number:
`type_error(number,Value)`
`Scale` is a variable:
`instantiation_error`
`Scale` is neither a variable nor a number:
`type_error(number,Scale)`
`Scale` is not positive:
`domain_error(positive_number,Scale)`

---

exponential_distribution/3

Computes the exponential cumulative distribution at the given value.

Compilation flags:
`static`

Template:
`exponential_distribution(Value,Scale,Probability)`
Mode and number of proofs:

```
exponential_distribution(+number,+positive_number,-float) - one_or_error
```

Exceptions:

`Value` is a variable:

```
instantiation_error
```

`Value` is neither a variable nor a number:

```
type_error(number,Value)
```

`Scale` is a variable:

```
instantiation_error
```

`Scale` is neither a variable nor a number:

```
type_error(number,Scale)
```

`Scale` is not positive:

```
domain_error(positive_number,Scale)
```

---

exponential_quantile/3

Computes the exponential quantile for a probability strictly between zero and one.

Compilation flags:

```
static
```

Template:

```
exponential_quantile(Probability,Scale,Quantile)
```

Mode and number of proofs:

```
exponential_quantile(+open_probability,+positive_number,-float) - one_or_error
```

Exceptions:

`Probability` is a variable:

```
instantiation_error
```

`Probability` is neither a variable nor a float:

```
type_error(float,Probability)
```

`Probability` is a float but not strictly between zero and one:

```
domain_error(open_probability,Probability)
```

`Scale` is a variable:

```
instantiation_error
```

`Scale` is neither a variable nor a number:

```
type_error(number,Scale)
```

`Scale` is not positive:

```
domain_error(positive_number,Scale)
```

---

fisher/3

Returns a Fisher-Snedecor (F) distributed random value.

Compilation flags:
`static`

Template:
`fisher(DegreesOfFreedom1,DegreesOfFreedom2,Value)`
Mode and number of proofs:
`fisher(+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:
`DegreesOfFreedom1` is a variable:
`instantiation_error`
`DegreesOfFreedom1` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom1)`
`DegreesOfFreedom1` is not positive:
`domain_error(positive_number,DegreesOfFreedom1)`
`DegreesOfFreedom2` is a variable:
`instantiation_error`
`DegreesOfFreedom2` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom2)`
`DegreesOfFreedom2` is not positive:
`domain_error(positive_number,DegreesOfFreedom2)`

---

fisher_samples/4

Returns the requested number of Fisher-Snedecor (F) distributed random values.

Compilation flags:
`static`

Template:
`fisher_samples(Count,DegreesOfFreedom1,DegreesOfFreedom2,Samples)`
Mode and number of proofs:
`fisher_samples(+integer,+positive_number,+positive_number,-list(float))` - `one_or_error`

Exceptions:
`Count` is a variable:
`instantiation_error`
`Count` is neither a variable nor an integer:
`type_error(integer,Count)`
`Count` is an integer but not a non-negative integer:
`domain_error(non_negative_integer,Count)`
`DegreesOfFreedom1` is a variable:

instantiation_error
DegreesOfFreedom1 is neither a variable nor a number:
type_error(number,DegreesOfFreedom1)
DegreesOfFreedom1 is not positive:
domain_error(positive_number,DegreesOfFreedom1)
DegreesOfFreedom2 is a variable:
instantiation_error
DegreesOfFreedom2 is neither a variable nor a number:
type_error(number,DegreesOfFreedom2)
DegreesOfFreedom2 is not positive:
domain_error(positive_number,DegreesOfFreedom2)

---

fisher_density/4

Computes the Fisher-Snedecor (F) probability density at the given value. At zero, returns the atom positive_infinity when the first degrees of freedom are smaller than two.

Compilation flags:
static

Template:
fisher_density(Value,DegreesOfFreedom1,DegreesOfFreedom2,Density)
Mode and number of proofs:
fisher_density(+number,+positive_number,+positive_number,-atomic) - one_or_error

Exceptions:
Value is a variable:
instantiation_error
Value is neither a variable nor a number:
type_error(number,Value)
DegreesOfFreedom1 is a variable:
instantiation_error
DegreesOfFreedom1 is neither a variable nor a number:
type_error(number,DegreesOfFreedom1)
DegreesOfFreedom1 is not positive:
domain_error(positive_number,DegreesOfFreedom1)
DegreesOfFreedom2 is a variable:
instantiation_error
DegreesOfFreedom2 is neither a variable nor a number:
type_error(number,DegreesOfFreedom2)
DegreesOfFreedom2 is not positive:
domain_error(positive_number,DegreesOfFreedom2)

---

fisher_log_density/4

Computes the Fisher-Snedecor (F) log-density at the given value. Returns the atoms positive_infinity or negative_infinity when the density is respectively infinite or zero.

Compilation flags:

static

Template:

fisher_log_density(Value,DegreesOfFreedom1,DegreesOfFreedom2,LogDensity)

Mode and number of proofs:

fisher_log_density(+number,+positive_number,+positive_number,-atomic) - one_or_error

Exceptions:

Value is a variable:

instantiation_error

Value is neither a variable nor a number:

type_error(number,Value)

DegreesOfFreedom1 is a variable:

instantiation_error

DegreesOfFreedom1 is neither a variable nor a number:

type_error(number,DegreesOfFreedom1)

DegreesOfFreedom1 is not positive:

domain_error(positive_number,DegreesOfFreedom1)

DegreesOfFreedom2 is a variable:

instantiation_error

DegreesOfFreedom2 is neither a variable nor a number:

type_error(number,DegreesOfFreedom2)

DegreesOfFreedom2 is not positive:

domain_error(positive_number,DegreesOfFreedom2)

---

fisher_distribution/4

Computes an approximation of the Fisher-Snedecor (F) cumulative distribution at the given value.

Compilation flags:

static

Template:

fisher_distribution(Value,DegreesOfFreedom1,DegreesOfFreedom2,Probability)

Mode and number of proofs:

fisher_distribution(+number,+positive_number,+positive_number,-float) - one_or_error

Exceptions:

Value is a variable:

instantiation_error

`Value` is neither a variable nor a number:
`type_error(number,Value)`
`DegreesOfFreedom1` is a variable:
`instantiation_error`
`DegreesOfFreedom1` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom1)`
`DegreesOfFreedom1` is not positive:
`domain_error(positive_number,DegreesOfFreedom1)`
`DegreesOfFreedom2` is a variable:
`instantiation_error`
`DegreesOfFreedom2` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom2)`
`DegreesOfFreedom2` is not positive:
`domain_error(positive_number,DegreesOfFreedom2)`

`fisher_quantile/4`

Computes an approximation of the Fisher-Snedecor (F) quantile for a probability strictly between zero and one.

Compilation flags:
`static`

Template:
`fisher_quantile(Probability,DegreesOfFreedom1,DegreesOfFreedom2,Quantile)`
Mode and number of proofs:
`fisher_quantile(+open_probability,+positive_number,+positive_number,-float)` - `one_or_error`

Exceptions:
`Probability` is a variable:
`instantiation_error`
`Probability` is neither a variable nor a float:
`type_error(float,Probability)`
`Probability` is a float but not strictly between zero and one:
`domain_error(open_probability,Probability)`
`DegreesOfFreedom1` is a variable:
`instantiation_error`
`DegreesOfFreedom1` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom1)`
`DegreesOfFreedom1` is not positive:
`domain_error(positive_number,DegreesOfFreedom1)`
`DegreesOfFreedom2` is a variable:
`instantiation_error`
`DegreesOfFreedom2` is neither a variable nor a number:
`type_error(number,DegreesOfFreedom2)`
`DegreesOfFreedom2` is not positive:
`domain_error(positive_number,DegreesOfFreedom2)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

univariate_distributions(Random), *multivariate_distributions_protocol*, *sampling_protocol*

1.345 uri_template

object

1.345.1 `uri_template(Representation)`

- `Representation` - URI template, variable name, variable value, and expansion representation. Valid values are `atom`, `codes`, and `chars`.

URI template validation, variable enumeration, and expansion predicates implementing RFC 6570.

Availability:
`logtalk_load(uri_template(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-09-24

Compilation flags:
`static, context_switching_calls`

Uses:
list
utf_8_character_set

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *valid/1*
  - *variables/2*
  - *expand/3*
  - *expand/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

valid/1

True iff the argument is a valid RFC 6570 URI template.

Compilation flags:
`static`

Template:
`valid(Template)`
Mode and number of proofs:
`valid(++text)` - `zero_or_one`

---

variables/2

Returns the unique template variable names in first-seen order.

Compilation flags:
`static`

Template:
`variables(Template,Variables)`
Mode and number of proofs:
`variables(++text,-list(text))` - `zero_or_one`

---

expand/3

Expands a valid RFC 6570 URI template using a list of `Name-Value` bindings. Values are represented by `undefined`, `string(Text)`, `list(Texts)`, `assoc(Pairs)`, or `structure(Pairs)` terms.

Compilation flags:
`static`

Template:

```
expand(Template,Bindings,Expansion)
```

Mode and number of proofs:

```
expand(++text,+list(pair),-text) - zero_or_one
```

---

expand/4

Expands an RFC 6570 URI template while recovering from template syntax errors. Malformed expressions are copied unexpanded and processing continues. For an invalid literal or unterminated expression, the unprocessed remainder is copied and processing stops. Errors are returned as zero-based `error(Position, Reason)` terms.

Compilation flags:

```
static
```

Template:

```
expand(Template,Bindings,Expansion,Errors)
```

Mode and number of proofs:

```
expand(++text,+list(pair),-text,-list(compound)) - zero_or_one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.346 url

object

1.346.1 url(Representation)

- `Representation` - URL and is components representation. Valid values are `atom`, `codes`, and `chars`.

URL validating, parsing, and normalizing predicates following RFC3986 nomenclature.

Availability:

```
logtalk_load(url(loader))
```

Author: Paulo Moura
Version: 2:0:0
Date: 2026-05-22

Compilation flags:
`static, context_switching_calls`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *valid/1*
  - *parse/2*
  - *generate/2*
  - *parse/3*
  - *reference_kind/2*
  - *equivalent/2*
  - *relativize/3*
  - *resolve/3*
  - *normalize/2*
  - *file_path_components/2*
- *Protected predicates*
- *Private predicates*
  - *downcase_text/2*
- *Operators*

Public predicates

valid/1

True iff the argument is a valid URL, including optional query and fragment components.

Compilation flags:
`static`

Template:
`valid(URL)`
Mode and number of proofs:
`valid(++text)` - `zero_or_one`

parse/2

Parses a URL into a list of its components: `[scheme(Scheme), authority(Authority), path(Path), query(Query), fragment(Fragment)]`. Fails if the URL is invalid and cannot be parsed.

Compilation flags:
`static`

Template:
`parse(URL,Components)`
Mode and number of proofs:
`parse(++text,-list(compound))` - `zero_or_one`

---

generate/2

Generates a normalized URL or RFC3986 relative reference from a list of components. Component lists headed by `scheme/1` generate absolute URLs; lists without `scheme/1` generate relative references.

Compilation flags:
`static`

Template:
`generate(Components,URL)`
Mode and number of proofs:
`generate(++list(compound),-text)` - `zero_or_one`

---

parse/3

Parses a URL or RFC3986 relative reference into a list of its components and classifies it as one of `url`, `network_path`, `absolute_path`, `relative_path`, `query`, or `fragment`.

Compilation flags:
`static`

Template:
`parse(Reference,Components,Kind)`
Mode and number of proofs:
`parse(++text,-list(compound),-atom)` - `zero_or_one`

---

reference_kind/2

Classifies a valid reference as one of `url`, `network_path`, `absolute_path`, `relative_path`, `query`, or `fragment`.

Compilation flags:
`static`

Template:
`reference_kind(Reference,Kind)`
Mode and number of proofs:
`reference_kind(++text,-atom)` - `zero_or_one`

---

equivalent/2

True iff both arguments normalize to the same textual representation under the library normalization rules.

Compilation flags:
`static`

Template:
`equivalent(Reference1,Reference2)`
Mode and number of proofs:
`equivalent(++text,++text)` - `zero_or_one`

---

relativize/3

Relativizes a target absolute hierarchical URL against a base absolute hierarchical URL with the same scheme and authority, returning a non-empty relative reference accepted by the library. Fails when no such non-empty relative reference can be generated.

Compilation flags:
`static`

Template:
`relativize(Base,Target,Relative)`
Mode and number of proofs:
`relativize(++text,++text,-text)` - `zero_or_one`

---

resolve/3

Resolves a reference against a base absolute hierarchical URL and returns a normalized absolute URL. The first argument must be an absolute hierarchical URL suitable for use as a base. The second argument may be either a relative reference or an absolute URL; when it is already absolute, the result is that absolute URL normalized and returned unchanged apart from normalization.

Compilation flags:

```
static
```

Template:

```
resolve(Base,Reference,Resolved)
```

Mode and number of proofs:

```
resolve(++text,++text,-text) - zero_or_one
```

---

normalize/2

Normalizes a URL or RFC3986 relative reference by standardizing its components. Normalization includes converting the scheme to lowercase when present, normalizing the authority, percent-encoding characters that require escaping, and simplifying path dot segments while preserving absolute and relative reference forms.

Compilation flags:

```
static
```

Template:

```
normalize(Reference,NormalizedReference)
```

Mode and number of proofs:

```
normalize(++text,-text) - one
```

---

file_path_components/2

Converts a file-system path into file URL components represented as `[authority(Authority), path(Path)]`. Windows drive-letter and UNC paths are normalized to RFC3986-compatible file URL components.

Compilation flags:

```
static
```

Template:

```
file_path_components(FilePath,Components)
```

Mode and number of proofs:

```
file_path_components(++text,-list(compound)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

`downcase_text/2`

Converts text to lowercase (ASCII only). Only uppercase letters A-Z are converted to lowercase.

Compilation flags:
`static`

Template:
`downcase_text(Text,LowerText)`
Mode and number of proofs:
`downcase_text(+text,-text) - one`

Operators

(none)

1.347 uuid

object

1.347.1 `uuid`

Universally unique identifier (UUID) generator using an atom representation.

Availability:
`logtalk_load(uuid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-24

Compilation flags:
`static, context_switching_calls`

Extends:
`public` *uuid(atom)*

Remarks:
(none)

Inherited public predicates:
random_node/1 *uuid_max/1* *uuid_nil/1* *uuid_null/1* *uuid_v1/2* *uuid_v1/3* *uuid_v3/3* *uuid_v4/1* *uuid_v5/3* *uuid_v7/1* *uuid_v7/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

uuid(Representation), *cuid2*, *ksuid*, *ids*, *nanoid*, *snowflakeid*, *typeid*, *ulid*

object

1.347.2 `uuid(Representation)`

- `Representation` - Text representation for the UUID. Possible values are `atom`, `chars`, and `codes`.

Universally unique identifier (UUID) generator.

Availability:
`logtalk_load(uuid(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-07-24

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *uuid_protocol*
Uses:
crypto
date
iso8601
list

os

Remarks:
(none)

Inherited public predicates:
random_node/1 *uuid_max/1* *uuid_nil/1* *uuid_null/1* *uuid_v1/2* *uuid_v1/3* *uuid_v3/3* *uuid_v4/1* *uuid_v5/3* *uuid_v7/1* *uuid_v7/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

uuid, *cuid2(Representation,Size,Alphabet)*, *ksuid(Representation,Alphabet)*, *ids(Representation,Bytes)*, *nanoid(Representation,Size,Alphabet)*, *snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceB*
typeid(Representation), *ulid(Representation)*

protocol

1.347.3 `uuid_protocol`

Universally unique identifier (UUID) generator protocol.

Availability:
`logtalk_load(uuid(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-07-06

Compilation flags:

`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *uuid_v1/2*
  - *uuid_v1/3*
  - *uuid_v3/3*
  - *uuid_v4/1*
  - *uuid_v5/3*
  - *uuid_v7/1*
  - *uuid_v7/2*
  - *uuid_null/1*
  - *uuid_nil/1*
  - *uuid_max/1*
  - *random_node/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

uuid_v1/2

Returns a version 1 UUID for the given MAC address (a list of six bytes). The MAC address can be replaced by a random 6 bytes node identifier as per RFC 4122 when the MAC address is not available or should not be disclosed. Portability note: some backends only provide access to local time; use `uuid_v1/3` with an explicit UTC offset when a UTC timestamp is required.

Compilation flags:
`static`

Template:
`uuid_v1(MAC,UUID)`
Mode and number of proofs:
`uuid_v1(+list(byte),--text)-one`

uuid_v1/3

Returns a version 1 UUID for the given MAC address (a list of six bytes) using the given local UTC offset (`Z` or `+HH:MM`/`-HH:MM`) to convert the backend local time to UTC. The MAC address can be replaced by a random 6 bytes node identifier as per RFC 4122 when the MAC address is not available or should not be disclosed.

Compilation flags:

```
static
```

Template:

```
uuid_v1(MAC,Offset,UUID)
```

Mode and number of proofs:

```
uuid_v1(+list(byte),+atom,--text) - zero_or_one
```

---

uuid_v3/3

Returns a version 3 UUID for the given namespace UUID and name. Namespace UUIDs and names can be represented as atoms, lists of characters, or lists of character codes. Name character codes must be bytes.

Compilation flags:

```
static
```

Template:

```
uuid_v3(Namespace,Name,UUID)
```

Mode and number of proofs:

```
uuid_v3(+text,+text,--text) - one
```

---

uuid_v4/1

Returns a version 4 UUID.

Compilation flags:

```
static
```

Template:

```
uuid_v4(UUID)
```

Mode and number of proofs:

```
uuid_v4(--text) - one
```

---

uuid_v5/3

Returns a version 5 UUID for the given namespace UUID and name. Namespace UUIDs and names can be represented as atoms, lists of characters, or lists of character codes. Name character codes must be bytes.

Compilation flags:
`static`

Template:
`uuid_v5(Namespace,Name,UUID)`
Mode and number of proofs:
`uuid_v5(+text,+text,--text)` - `one`

---

uuid_v7/1

Returns a version 7 UUID. Portability note: some backends only provide access to local time; use `uuid_v7/2` with an explicit UTC offset when a UTC timestamp is required.

Compilation flags:
`static`

Template:
`uuid_v7(UUID)`
Mode and number of proofs:
`uuid_v7(--text)` - `one`

---

uuid_v7/2

Returns a version 7 UUID using the given local UTC offset (`Z` or `+HH:MM`/`-HH:MM`) to convert the backend local time to UTC.

Compilation flags:
`static`

Template:
`uuid_v7(Offset,UUID)`
Mode and number of proofs:
`uuid_v7(+atom,--text)` - `zero_or_one`

---

uuid_null/1

Returns the null UUID. Deprecated. Use `uuid_nil/1` instead.

Compilation flags:
`static`

Template:
`uuid_null(UUID)`
Mode and number of proofs:
`uuid_null(--text)` - `one`

---

uuid_nil/1

Returns the Nil UUID.

Compilation flags:
`static`

Template:
`uuid_nil(UUID)`
Mode and number of proofs:
`uuid_nil(--text)` - `one`

---

uuid_max/1

Returns the Max UUID.

Compilation flags:
`static`

Template:
`uuid_max(UUID)`
Mode and number of proofs:
`uuid_max(--text)` - `one`

`random_node/1`

Generates a list with six random bytes that can be used in alternative to a MAC address when generating version 1 UUIDs.

Compilation flags:
`static`

Template:
`random_node(Node)`
Mode and number of proofs:
`random_node(--list(byte))` - `one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.348 validations

object

1.348.1 `validated`

Types and predicates for type-checking and handling lists of validation terms. Inspired by Scala Cats and Kotlin Arrow.

Availability:
`logtalk_load(validations(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-02-22

Compilation flags:
`static, context_switching_calls`

Provides:
type::type/1
type::check/2
arbitrary::arbitrary/1
arbitrary::arbitrary/2
arbitrary::shrinker/1

arbitrary::shrink/3
arbitrary::edge_case/2

Uses:

list
random
type

Remarks:

- Type-checking support: Defines a `validated(ValueType, ErrorType)` type for checking validation terms where the value and error terms must be of the given types.
- QuickCheck support: Defines clauses for the `type::arbitrary/1-2`, `arbitrary::shrinker/1`, `arbitrary::shrink/3`, and `arbitrary::edge_case/2` predicates to allow generating random values for the `validated(ValueType, ErrorType)` type.

Inherited public predicates:

(none)

- *Public predicates*
  - *valids/2*
  - *invalids/2*
  - *partition/3*
  - *map/3*
  - *map/4*
  - *sequence/2*
  - *traverse/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

valids/2

Returns the values stored in validation terms that hold valid values.

Compilation flags:

`static`

Template:

`valids(Validations,Values)`

Mode and number of proofs:

`valids(+list(validation),-list)` - `one`

invalids/2

Returns a flattened list with all errors stored in invalid validation terms.

Compilation flags:
```
static
```

Template:
```
invalids(Validations,Errors)
```
Mode and number of proofs:
```
invalids(+list(validation),-list) - one
```

---

partition/3

Retrieves and partitions valid values and flattened accumulated errors from validation terms.

Compilation flags:
```
static
```

Template:
```
partition(Validations,Values,Errors)
```
Mode and number of proofs:
```
partition(+list(validation),-list,-list) - one
```

---

map/3

Applies a closure to each list element to generate validation terms and returns a pair `Values-Errors` accumulating all valid values and all errors in one pass.

Compilation flags:
```
static
```

Template:
```
map(Closure,Terms,ValuesErrors)
```
Meta-predicate template:
```
map(2,*,*)
```
Mode and number of proofs:
```
map(+callable,+list,--compound) - one
```

---

map/4

Applies a closure to each list element to generate validation terms and returns valid values and accumulated errors in one pass.

Compilation flags:
```
static
```

Template:
```
map(Closure,Terms,Values,Errors)
```
Meta-predicate template:
```
map(2,*,*,*)
```
Mode and number of proofs:
```
map(+callable,+list,-list,-list) - one
```

---

sequence/2

Sequences a list of validation terms into a single validation term, accumulating all errors.

Compilation flags:
```
static
```

Template:
```
sequence(Validations,Validation)
```
Mode and number of proofs:
```
sequence(+list(validation),--nonvar) - one
```

---

traverse/3

Applies a closure to each list element to generate validation terms and then sequences them, accumulating all errors.

Compilation flags:
```
static
```

Template:
```
traverse(Closure,Terms,Validation)
```
Meta-predicate template:
```
traverse(2,*,*)
```
Mode and number of proofs:
```
traverse(+callable,+list,--nonvar) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

validation, *validation(Validation)*, *type*, *arbitrary*

object

1.348.2 `validation`

Constructors for validation terms. A validation term is either `valid(Value)` or `invalid(Errors)` where `Errors` is a list of errors. Validation terms allow applicative-style error accumulation.

Availability:
`logtalk_load(validations(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-04-22

Compilation flags:
`static, context_switching_calls`

Provides:
type::type/1
type::check/2
Uses:
type

Remarks:

- Type-checking support: Defines a `validation` type for use with the `type` library object.

Inherited public predicates:
(none)

- *Public predicates*
  - *of_valid/2*
  - *of_invalid/2*

      - *of_invalids/2*
      - *from_goal/4*
      - *from_goal/3*
      - *from_goal/2*
      - *from_generator/4*
      - *from_generator/3*
      - *from_generator/2*
      - *from_optional/3*
      - *from_expected/2*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

of_valid/2

Constructs a validation term holding a valid value.

Compilation flags:
`static`

Template:
`of_valid(Value,Validation)`
Mode and number of proofs:
`of_valid(@term,--nonvar)` - `one`

---

of_invalid/2

Constructs a validation term holding a single error.

Compilation flags:
`static`

Template:
`of_invalid(Error,Validation)`
Mode and number of proofs:
`of_invalid(@term,--nonvar)` - `one`

---

of_invalids/2

Constructs a validation term holding a list of errors.

Compilation flags:
`static`

Template:
`of_invalids(Errors,Validation)`
Mode and number of proofs:
`of_invalids(@list,--nonvar)` - `one`

---

from_goal/4

Constructs a validation term holding a value bound by calling the given goal. Otherwise returns a validation term with a single error represented by the `Error` argument.

Compilation flags:
`static`

Template:
`from_goal(Goal,Value,Error,Validation)`
Meta-predicate template:
`from_goal(0,*,*,*)`
Mode and number of proofs:
`from_goal(+callable,--term,@term,--nonvar)` - `one`

---

from_goal/3

Constructs a validation term holding a value bound by calling the given goal. Otherwise returns a validation term with a single error being the goal error or the atom `fail` representing goal failure.

Compilation flags:
`static`

Template:
`from_goal(Goal,Value,Validation)`
Meta-predicate template:
`from_goal(0,*,*)`
Mode and number of proofs:
`from_goal(+callable,--term,--nonvar)` - `one`

---

from_goal/2

Constructs a validation term holding a value bound by calling the given closure. Otherwise returns a validation term with a single error being the closure error or the atom `fail` representing closure failure.

Compilation flags:
`static`

Template:
`from_goal(Closure,Validation)`
Meta-predicate template:
`from_goal(1,*)`
Mode and number of proofs:
`from_goal(+callable,--nonvar)` - `one`

---

from_generator/4

Constructs validation terms with the values generated by calling the given goal. On goal error or failure, returns a validation term with a single error represented by the `Error` argument.

Compilation flags:
`static`

Template:
`from_generator(Goal,Value,Error,Validation)`
Meta-predicate template:
`from_generator(0,*,*,*)`
Mode and number of proofs:
`from_generator(+callable,--term,@term,--nonvar)` - `one_or_more`

---

from_generator/3

Constructs validation terms with the values generated by calling the given goal. On goal error or failure, returns a validation term with, respectively, a single error being the goal error or the atom `fail` representing goal failure.

Compilation flags:
`static`

Template:
`from_generator(Goal,Value,Validation)`
Meta-predicate template:
`from_generator(0,*,*)`
Mode and number of proofs:
`from_generator(+callable,--term,--nonvar)` - `one_or_more`

---

from_generator/2

Constructs validation terms with the values generated by calling the given closure. On closure error or failure, returns a validation term with, respectively, a single error being the closure error or the atom `fail` representing closure failure.

Compilation flags:
`static`

Template:
`from_generator(Closure,Validation)`
Meta-predicate template:
`from_generator(1,*)`
Mode and number of proofs:
`from_generator(+callable,--nonvar)` - `one_or_more`

---

from_optional/3

Converts an optional term to a validation term. Returns a valid term holding the value if the optional term is not empty. Returns an invalid term with the given error otherwise.

Compilation flags:
`static`

Template:
`from_optional(Optional,Error,Validation)`
Mode and number of proofs:
`from_optional(+nonvar,@term,--nonvar)` - `one`

---

from_expected/2

Converts an expected term to a validation term. Returns a valid term holding the value if the expected term holds a value. Returns an invalid term with the expected term error otherwise.

Compilation flags:
`static`

Template:
`from_expected(Expected,Validation)`
Mode and number of proofs:
`from_expected(+nonvar,--nonvar)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

validation(Validation), *validated*, *type*, *arbitrary*

object

1.348.3 `validation(Validation)`

Validation term predicates. Requires passing a validation term (constructed using the `validation` object predicates) as a parameter.

Availability:
`logtalk_load(validations(loader))`

Author: Paulo Moura
Version: 0:2:0
Date: 2026-02-21

Compilation flags:
`static, context_switching_calls`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *is_valid/0*
  - *is_invalid/0*
  - *if_valid/1*
  - *if_invalid/1*
  - *if_valid_or_else/2*
  - *valid/1*

      - *invalid/1*
      - *filter/3*
      - *map/2*
      - *flat_map/2*
      - *map_or_else/3*
      - *map_catching/2*
      - *map_invalid/2*
      - *map_both/3*
      - *swap/1*
      - *or/2*
      - *or_else/2*
      - *or_else_get/2*
      - *or_else_call/2*
      - *or_else_fail/1*
      - *or_else_throw/1*
      - *or_else_throw/2*
      - *zip/3*
      - *flatten/1*
      - *to_optional/1*
      - *to_expected/1*
  - *Protected predicates*
  - *Private predicates*
  - *Operators*

Public predicates

is_valid/0

True if the validation term holds a valid value.

Compilation flags:

```
static
```

Mode and number of proofs:

```
is_valid - zero_or_one
```

---

is_invalid/0

True if the validation term holds one or more errors.

Compilation flags:

```
static
```

Mode and number of proofs:

```
is_invalid - zero_or_one
```

---

if_valid/1

Applies a closure when the validation term holds a valid value using the value as argument. Succeeds otherwise.

Compilation flags:
`static`

Template:
`if_valid(Closure)`
Meta-predicate template:
`if_valid(1)`
Mode and number of proofs:
`if_valid(+callable)` - `zero_or_more`

---

if_invalid/1

Applies a closure when the validation term holds errors using the errors list as argument. Succeeds otherwise.

Compilation flags:
`static`

Template:
`if_invalid(Closure)`
Meta-predicate template:
`if_invalid(1)`
Mode and number of proofs:
`if_invalid(+callable)` - `zero_or_more`

---

if_valid_or_else/2

Applies either `ValidClosure` or `InvalidClosure` depending on the validation term holding a value or errors.

Compilation flags:
`static`

Template:
`if_valid_or_else(ValidClosure,InvalidClosure)`
Meta-predicate template:
`if_valid_or_else(1,1)`
Mode and number of proofs:
`if_valid_or_else(+callable,+callable)` - `zero_or_more`

---

valid/1

Returns the value hold by the validation term. Throws an error otherwise.

Compilation flags:
`static`

Template:
`valid(Value)`
Mode and number of proofs:
`valid(--term)` - `one_or_error`

Exceptions:
Validation term holds errors:
`existence_error(valid_value,Validation)`

---

invalid/1

Returns the errors hold by the validation term. Throws an error otherwise.

Compilation flags:
`static`

Template:
`invalid(Errors)`
Mode and number of proofs:
`invalid(--list)` - `one_or_error`

Exceptions:
Validation term holds a valid value:
`existence_error(validation_errors,Validation)`

---

filter/3

When the validation term holds a value and the value satisfies the closure, returns the same validation term. When the validation term holds a value that does not satisfy the closure, returns an invalid term with the given error. When the validation term holds errors, returns the same validation term.

Compilation flags:
`static`

Template:
`filter(Closure,Error,NewValidation)`
Meta-predicate template:
`filter(1,*,*)`

Mode and number of proofs:

```
filter(+callable,@term,--nonvar) - one
```

---

map/2

When the validation term holds a valid value and mapping a closure with the value and the new value as additional arguments succeeds, returns a new valid term. Otherwise returns the same validation term.

Compilation flags:

```
static
```

Template:

```
map(Closure,NewValidation)
```

Meta-predicate template:

```
map(2,*)
```

Mode and number of proofs:

```
map(+callable,--nonvar) - one
```

---

flat_map/2

When the validation term holds a valid value, applies a closure with the value and the new validation term as additional arguments. Returns the new validation term on success. When the validation term holds errors, short-circuits by returning the same validation term without calling the closure. This is the monadic escape hatch for dependent steps; use `zip/3`, `sequence/2`, or `traverse/3` for error accumulation on independent steps.

Compilation flags:

```
static
```

Template:

```
flat_map(Closure,NewValidation)
```

Meta-predicate template:

```
flat_map(2,*)
```

Mode and number of proofs:

```
flat_map(+callable,--nonvar) - one
```

---

map_or_else/3

When the validation term holds a value and mapping a closure with the value and the new value as additional arguments is successful, returns the new value. Otherwise returns the given default value.

Compilation flags:

```
static
```

Template:

```
map_or_else(Closure,Default,Value)
```

Meta-predicate template:

```
map_or_else(2,*,*)
```

Mode and number of proofs:

```
map_or_else(+callable,@term,--term) - one
```

---

map_catching/2

When the validation term holds a value, applies a closure to it. Returns a valid term with the new value if the closure succeeds. Returns an invalid term with the error if the closure throws an error. Returns an invalid term with the atom `fail` as error if the closure fails. When the validation term holds errors, returns the same validation term.

Compilation flags:

```
static
```

Template:

```
map_catching(Closure,NewValidation)
```

Meta-predicate template:

```
map_catching(2,*)
```

Mode and number of proofs:

```
map_catching(+callable,--nonvar) - one
```

---

map_invalid/2

When the validation term holds errors and mapping a closure with the errors list and the new errors list as additional arguments succeeds, returns a new invalid term. Otherwise returns the same validation term.

Compilation flags:

```
static
```

Template:

```
map_invalid(Closure,NewValidation)
```

Meta-predicate template:

```
map_invalid(2,*)
```

Mode and number of proofs:

```
map_invalid(+callable,--nonvar) - one
```

---

map_both/3

When the validation term holds a value and mapping `ValidClosure` with the value is successful, returns a valid term with the new value. When the validation term holds errors and mapping `InvalidClosure` with the errors list is successful, returns an invalid term with the new errors. Otherwise returns the same validation term.

Compilation flags:

```
static
```

Template:

```
map_both(ValidClosure,InvalidClosure,NewValidation)
```

Meta-predicate template:

```
map_both(2,2,*)
```

Mode and number of proofs:

```
map_both(+callable,+callable,--nonvar) - one
```

---

swap/1

Swaps the valid and invalid terms. If the validation term holds a value, returns an invalid term with a singleton list containing that value. If the validation term holds errors, returns a valid term with the errors list.

Compilation flags:

```
static
```

Template:

```
swap(NewValidation)
```

Mode and number of proofs:

```
swap(--nonvar) - one
```

---

or/2

Returns the same validation term if it holds a value. Otherwise calls closure to generate a new validation term. Fails if the validation term holds errors and calling the closure fails or throws an error.

Compilation flags:

```
static
```

Template:

```
or(NewValidation,Closure)
```

Meta-predicate template:

```
or(*,1)
```

Mode and number of proofs:

```
or(--term,@callable) - zero_or_one
```

---

or_else/2

Returns the valid value if present or the default value otherwise.

Compilation flags:
```
static
```

Template:
```
or_else(Value,Default)
```
Mode and number of proofs:
```
or_else(--term,@term) - one
```

---

or_else_get/2

Returns the value hold by the validation term if valid. Otherwise applies a closure to compute the value. Throws an error when the validation term holds errors and a value cannot be computed.

Compilation flags:
```
static
```

Template:
```
or_else_get(Value,Closure)
```
Meta-predicate template:
```
or_else_get(*,1)
```
Mode and number of proofs:
```
or_else_get(--term,+callable) - one_or_error
```

Exceptions:
Validation term holds errors and a value cannot be computed:
```
existence_error(valid_value,Validation)
```

---

or_else_call/2

Returns the value hold by the validation term if valid. Calls a goal deterministically otherwise.

Compilation flags:
```
static
```

Template:
```
or_else_call(Value,Goal)
```
Meta-predicate template:
```
or_else_call(*,0)
```
Mode and number of proofs:
```
or_else_call(--term,+callable) - zero_or_one
```

---

or_else_fail/1

Returns the valid value if present. Fails otherwise.

Compilation flags:
`static`

Template:
`or_else_fail(Value)`
Mode and number of proofs:
`or_else_fail(--term)` - `zero_or_one`

---

or_else_throw/1

Returns the value hold by the validation term if valid. Throws the errors list hold by the validation term as an exception otherwise.

Compilation flags:
`static`

Template:
`or_else_throw(Value)`
Mode and number of proofs:
`or_else_throw(--term)` - `one_or_error`

---

or_else_throw/2

Returns the value hold by the validation term if valid. Throws the given error otherwise, ignoring any errors hold by the validation term.

Compilation flags:
`static`

Template:
`or_else_throw(Value,Error)`
Mode and number of proofs:
`or_else_throw(--term,@nonvar)` - `one_or_error`

---

zip/3

When both this validation and the other validation hold values and applying a closure with both values and the new value as additional arguments is successful, returns a valid term with the new value. When both hold errors, returns an invalid term with all errors accumulated. Otherwise returns the first invalid term.

Compilation flags:

```
static
```

Template:

```
zip(Closure,OtherValidation,NewValidation)
```

Meta-predicate template:

```
zip(3,*,*)
```

Mode and number of proofs:

```
zip(+callable,+nonvar,--nonvar) - one
```

---

flatten/1

Flattens a nested validation term. When the validation term holds a value that is itself a validation term, returns the inner validation term. When the validation term holds a non-validation value, returns the same validation term. When the validation term holds errors, returns the same validation term.

Compilation flags:

```
static
```

Template:

```
flatten(NewValidation)
```

Mode and number of proofs:

```
flatten(--nonvar) - one
```

---

to_optional/1

Converts the validation term to an optional term. Returns a non-empty optional term holding the value if the validation term holds a value. Returns an empty optional term if the validation term holds errors.

Compilation flags:

```
static
```

Template:

```
to_optional(Optional)
```

Mode and number of proofs:

```
to_optional(--nonvar) - one
```

---

to_expected/1

Converts the validation term to an expected term. Returns an expected term holding the value if the validation term holds a value. Returns an expected term with the errors list as the unexpected error otherwise.

Compilation flags:
`static`

Template:
`to_expected(Expected)`
Mode and number of proofs:
`to_expected(--nonvar)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

validation, *validated*

1.349 verdi_neruda

object

1.349.1 a_star_interpreter(W)

A* interpreter for general logic programs. The parameter W is used to fine tune the behavior. W = 0 gives us a breadth-first search and W = 1 gives us a greedy best-first search. The default value for W is 0.5.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:0
Date: 2010-06-13

Compilation flags:
`static, context_switching_calls`

Imports:

`public` *best_first*

Remarks:
(none)

Inherited public predicates:
prove/2 *prove/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.349.2 `benchmark_generators`

Generates random data structures for use in benchmarks.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:0
Date: 2010-06-13

Compilation flags:
`static, context_switching_calls`

Uses:
random

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *random_tree/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

random_tree/1

Generates a random tree.

Compilation flags:
`static`

Template:
`random_tree(Tree)`
Mode and number of proofs:
`random_tree(-tree)` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.349.3 best_first

Best-first framework for general logic programs.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:1:1
Date: 2026-01-28

Compilation flags:
`static`

Implements:
`public` *interpreterp*

Uses:
counter
pairing_heap(Order)

Remarks:
(none)

Inherited public predicates:
prove/2 *prove/3*

- *Public predicates*
- *Protected predicates*
  - *f/4*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

f/4

.

Compilation flags:
`static`

Template:
`f(Length1,Length2,Depth,Cost)`

Mode and number of proofs:
`f(+float,+float,+float,-float)` - `zero_or_more`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.349.4 bfs_interpreter

Breadth-first interpreter for general logic programs.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:0
Date: 2010-06-13

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *interpreterp*
Uses:
counter
queue

Remarks:
(none)

Inherited public predicates:
prove/2 *prove/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.349.5 `bup_interpreter`

Semi-naive bottom-up interpreter for general (stratified) logic programs. Magic transformation is realized through an expansion hook.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Ulf Nilsson. Ported to Logtalk and augmented with negation by Victor Lagerkvist.
Version: 1:1:3
Date: 2023-11-30

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *interpreterp*
Uses:
counter
list
magic
term

Remarks:
(none)

Inherited public predicates:
prove/2 *prove/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.349.6 `counter`

Counter implemented with asserta/retract.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:1
Date: 2022-10-08

Compilation flags:
`static, context_switching_calls`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *increment/0*
  - *increase/1*
  - *set/1*
  - *value/1*
  - *reset/0*
- *Protected predicates*
- *Private predicates*
  - *c/1*
- *Operators*

Public predicates

increment/0

Increment the counter by 1.

Compilation flags:
`static`

Mode and number of proofs:
`increment` - `one`

---

increase/1

Increments the counter by the specified amount.

Compilation flags:
`static`

Template:
`increase(I)`
Mode and number of proofs:
`increase(+number)` - `one`

---

set/1

Sets the counter to the specified amount.

Compilation flags:
`static`

Template:
`set(N)`
Mode and number of proofs:
`set(+number)` - `one`

---

value/1

Gets the current value of the counter.

Compilation flags:
`static`

Template:
`value(N)`
Mode and number of proofs:
`value(?number)` - `one`

`reset/0`

Resets the counter to zero.

Compilation flags:
`static`

Mode and number of proofs:
`reset` - `one`

Protected predicates

(none)

Private predicates

`c/1`

Stores the current value of the counter.

Compilation flags:
`dynamic`

Template:
`c(N)`
Mode and number of proofs:
`c(?number)` - `zero_or_one`

Operators

(none)
protocol

1.349.7 databasep

Database protocol.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:0
Date: 2010-06-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *rule/4*
  - *rule/3*
  - *rule/2*
  - *bench_goal/1*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

rule/4

Clauses for this predicate are automatically generated using term-expansion. The third argument contains the length of Body.

Compilation flags:
`static`

Template:
`rule(Head,Body,Length,Tail)`
Mode and number of proofs:
`rule(?callable,?callable,-,-)` - `zero_or_more`

rule/3

Clauses for this predicate are automatically generated using term-expansion. The third argument denotes the tail of the Body.

Compilation flags:
`static`

Template:
`rule(Head,Body,Tail)`
Mode and number of proofs:
`rule(?callable,?callable,-)` - `zero_or_more`

---

rule/2

Clauses for this predicate are automatically generated using term-expansion.

Compilation flags:
`static`

Template:
`rule(Head,Body)`
Mode and number of proofs:
`rule(?callable,-list(callable))` - `zero_or_more`

---

bench_goal/1

Table of benchmark goals. They are used from shell.lgt to make benchmarking easier.

Compilation flags:
`static`

Template:
`bench_goal(Goal)`
Mode and number of proofs:
`bench_goal(?callable)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.349.8 `debug_expansion(Mode)`

Expands debug/1 calls. The parameter Mode can be either the atom “debug” or “production”.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2010-04-15

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.349.9 demodb

Sample database of simple test programs.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:0
Date: 2026-08-10

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *databasep*

Remarks:
(none)

Inherited public predicates:
bench_goal/1 *rule/2* *rule/3* *rule/4*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.349.10 `dfs_interpreter`

Depth-first interpreter for general logic programs.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:0
Date: 2010-06-13

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *interpreterp*
Uses:
counter

Remarks:
(none)

Inherited public predicates:
prove/2 *prove/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
category

1.349.11 `flatting`

Flattens conjunction of goals with the form f and g into a list [f,g].

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:1:0
Date: 2025-10-06

Compilation flags:
`static`

source: Based on source code from The Craft of Prolog, by Richard O'Keefe.

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
- *Protected predicates*
  - *flatten_goals//1*
- *Private predicates*
- *Operators*

Public predicates

(none)

Protected predicates

`flatten_goals//1`

Flattens a conjunction of goals.

Compilation flags:
`static`

Template:
`flatten_goals(Conjunction)`
Mode and number of proofs:
`flatten_goals(+callable)` - `one`

Private predicates

(none)

Operators

(none)
object

1.349.12 `heuristic_expansion(Mode)`

Expands rules of the form p if f and g to rule(p, [f,g|Tail], Length, Tail).

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:2
Date: 2022-10-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Extends:
`public` *rule_expansion(Mode)*
Uses:
list

Remarks:

(none)

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.349.13 `iddfs_interpreter(Increment)`

Iterative deepening depth-first interpreter for general logic programs. Based on source code from The Craft of Prolog, by Richard O'Keefe. The default value for the increment is 1.

Availability:

```
logtalk_load(verdi_neruda(loader))
```

Author: Victor Lagerkvist
Version: 1:0:0
Date: 2010-06-13

Compilation flags:

```
static, context_switching_calls
```

Implements:

`public` *interpreterp*

Uses:

counter
dfs_interpreter

Remarks:

(none)

Inherited public predicates:
prove/2 *prove/3*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.349.14 `interpreterp`

Protocol for an interpreter.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:0
Date: 2010-06-13

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *prove/2*
  - *prove/3*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

prove/2

True if goal is provable in the specified database.

Compilation flags:
`static`

Template:
`prove(Goal,DB)`
Mode and number of proofs:
`prove(+goal,+database)` - `zero_or_more`

---

prove/3

True if goal is provable within the given depth-limit in the specified database.

Compilation flags:
`static`

Template:
`prove(Goal,Limit,DB)`
Mode and number of proofs:
`prove(+goal,+limit,+database)` - `zero_or_more`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)
object

1.349.15 `magic`

Object encapsulating magic methods.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Ulf Nilsson. Ported to Logtalk and augmented with stratified negation by Victor Lagerkvist.
Version: 1:0:0
Date: 2010-06-13

Compilation flags:
`static, context_switching_calls`

Uses:
list

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *magicise/4*
  - *magic/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`magicise/4`

Transform (Head :- Body) into a magic clause (NewHead :- NewBody).

Compilation flags:
`static`

Template:

```
magicise(Head,Body,NewHead,NewBody)
```
Mode and number of proofs:
```
magicise(+term,+list,-term,-list) - zero_or_one
```

---

magic/2

Prefix the predicate symbol of Old with *magic*.

Compilation flags:
```
static
```

Template:
```
magic(Old,New)
```
Mode and number of proofs:
```
magic(+callable,-callable) - zero_or_one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.349.16 magic_expansion(Mode)

Expands rules of the form p if f and g to the more manageable rule(p, [f,g]) and performs magic transformation of clauses.

Availability:
```
logtalk_load(verdi_neruda(loader))
```

Author: Victor Lagerkvist
Version: 1:0:2
Date: 2022-10-08

Compilation flags:
```
static, context_switching_calls
```

Implements:
```
public expanding
```
Imports:

`public` *flatting*

Extends:

`public` *debug_expansion(Mode)*

Uses:

list

magic

Remarks:

(none)

Inherited public predicates:

goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.349.17 `rule_expansion(Mode)`

Expands rules of the form p if f and g to the more manageable rule(p, [f,g]).

Availability:

`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:2
Date: 2022-10-08

Compilation flags:

`static, context_switching_calls`

Implements:
`public` *expanding*
Imports:
`public` *flatting*
Extends:
`public` *debug_expansion(Mode)*

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.349.18 `shell`

User frontend to start the application.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-08-10

Compilation flags:
`static, context_switching_calls`

Uses:
shell(Interpreters)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - `welcome/0`
  - `start/0`
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`welcome/0`

Prints a welcome message.

Compilation flags:
`static`

Mode and number of proofs:
`welcome` - `one`

`start/0`

Starts the application shell with the available interpreters.

Compilation flags:
`static`

Mode and number of proofs:
`start` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.349.19 `shell(Interpreters)`

Prolog shell for the interpreters.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist and Paulo Moura
Version: 1:2:0
Date: 2026-08-10

Compilation flags:
`static, context_switching_calls`

Uses:
counter
list
meta
pairs

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *init/0*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

`init/0`

Starts the read-eval-print loop for the interpreters.

Compilation flags:
`static`

Mode and number of proofs:
`init` - `one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
object

1.349.20 `shell_expansion(Mode)`

Expansion object for the shell.

Availability:
`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist
Version: 1:0:1
Date: 2022-10-08

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Extends:
`public` *rule_expansion(Mode)*

Remarks:
(none)

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.350 wkt_wkb

object

1.350.1 `wkt_wkb`

Well-Known Text (WKT) and Well-Known Binary (WKB) geometry parser, generator, and validator.

Availability:
```
logtalk_load(wkt_wkb(loader))
```

Author: Paulo Moura
Version: 1:2:0
Date: 2026-09-25

Compilation flags:
```
static, context_switching_calls
```

Implements:
`public` *wkt_wkb_protocol*
Uses:
crypto
ieee_754(Precision,ByteOrder,NaNRepresentation)
ieee_754_fields(Precision,ByteOrder)
list
reader

Remarks:

(none)

Inherited public predicates:
generate/2 *parse/2* *validate/1* *validate/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

wkt_wkb_protocol, *geojson*, *geospatial*, *cbor*, *message_pack(StringRepresentation)*

protocol

1.350.2 `wkt_wkb_protocol`

Well-Known Text (WKT) and Well-Known Binary (WKB) geometry parser, generator, and validator protocol.

Availability:
`logtalk_load(wkt_wkb(loader))`

Author: Paulo Moura
Version: 1:0:1
Date: 2026-06-15

Compilation flags:
`static`

Dependencies:
(none)

Remarks:

(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *generate/2*
  - *validate/1*
  - *validate/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses a geometry from the given source. Supported sources are `wkt(file(Path))`, `wkt(stream(Stream))`, `wkt(atom(Atom))`, `wkt(chars(List))`, `wkt(codes(List))`, `wkb(file(Path))`, `wkb(stream(Stream))`, `wkb(bytes(List))`, `wkb(hex(atom(Atom)))`, `wkb(hex(chars(List)))`, and `wkb(hex(codes(List)))`.

Compilation flags:
`static`

Template:
`parse(Source,Geometry)`

Mode and number of proofs:
`parse(++compound,--term)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(wkt_wkb_source,Source)`
`Source` data does not represent a valid WKT or WKB geometry:
`domain_error(wkt_wkb,Source)`

generate/2

Generates WKT or WKB content using the representation specified in the first argument from a native geometry term. Supported sinks are `wkt(file(Path))`, `wkt(stream(Stream))`, `wkt(atom(Atom))`, `wkt(chars(List))`, `wkt(codes(List))`, `wkb(file(Path))`, `wkb(stream(Stream))`, `wkb(bytes(List))`, `wkb(hex(atom(Atom)))`, `wkb(hex(chars(List)))`, `wkb(hex(codes(List)))`, and the same WKB sink wrappers wrapped as `wkb(Sink, Order)` where `Order` is `little` or `big`.

Compilation flags:

`static`

Template:

`generate(Sink,Geometry)`

Mode and number of proofs:

`generate(+compound,++term)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`Sink` is neither a variable nor a valid sink:

`domain_error(wkt_wkb_sink,Sink)`

`Geometry` is not a valid native geometry term:

`domain_error(wkt_wkb,Geometry)`

`Geometry` is valid but cannot be represented in WKT due to special ordinates:

`domain_error(wkt_wkb_representation,wkt(Geometry))`

---

validate/1

True iff the argument is a valid native WKT/WKB geometry term.

Compilation flags:

`static`

Template:

`validate(Geometry)`

Mode and number of proofs:

`validate(@term)` - `zero_or_one`

---

validate/2

Validates a native geometry term returning a list of reason terms whose final argument is the failing path.

Compilation flags:

`static`

Template:

`validate(Geometry,Errors)`

Mode and number of proofs:

`validate(@term,-list(compound))` - `one_or_error`

Exceptions:

`Geometry` is a variable:

`instantiation_error`

Remarks:

- invalid_geometry_term(Path): The term is not any supported geometry representation.
- invalid_options(Path): The options argument is not a list.
- unknown_option(Option, Path): An option term is not recognized.
- duplicate_option(Name, Path): The same option name occurs more than once.
- invalid_dimensions(Path): A `dimensions/1` option does not use one of the supported values `xy`, `z`, `m`, or `zm`.
- invalid_position(Path): A position is not a list of numeric or supported special IEEE-754 coordinates with the expected dimensionality.
- invalid_position_array(Path): A coordinate array expected to contain positions is not a list.
- mixed_coordinate_dimension(Path): Coordinates in the same geometry do not all have the same dimensionality.
- coordinate_dimension_mismatch(Actual, Expected, Path): The coordinates dimensionality does not match the `dimensions/1` option.
- insufficient_positions(MinimumLength, Path): A coordinate array contains fewer positions than required for the enclosing geometry.
- ring_not_closed(Path): A polygon ring does not repeat its first position as its last position.
- invalid_polygon(Path): A polygon coordinate structure is invalid.
- invalid_multi_polygon(Path): A multi-polygon coordinate structure is invalid.
- invalid_geometry_collection(Path): A geometry collection member is not a list of valid geometry terms.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

wkt_wkb, *geojson*, *geospatial*

1.351 wrapper

object

1.351.1 `wrapper`

Adviser tool for porting and wrapping plain Prolog applications.

Availability:
`logtalk_load(wrapper(loader))`

Author: Paulo Moura
Version: 0:12:4
Date: 2026-01-20

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *expanding*
Provides:
logtalk::message_hook/4
logtalk::message_prefix_stream/4
logtalk::message_tokens//2
Uses:
logtalk
os

Remarks:

- `prolog_extensions(Extensions)` option: List of file name extensions used to recognize Prolog source files (default is `['.pl','.pro','.prolog']`).
- `logtalk_extension(Extension)` option: Logtalk file name extension to be used for the generated wrapper files (default is `'.lgt'`).
- `exclude_files(Files)` option: List of Prolog source files names to exclude (default is `[]`).
- `exclude_directories(Directories)` option: List of sub-directory names to exclude (default is `[]`).
- `include_wrapped_files(Boolean)`: Generate `include/1` directives for the wrapped Prolog source files (default is `true`).

Inherited public predicates:
goal_expansion/2 *term_expansion/2*

- *Public predicates*
  - *rdirectory/2*
  - *rdirectory/1*
  - *directory/2*
  - *directory/1*
  - *directories/2*
  - *directories/1*
  - *files/2*
  - *files/1*
  - *file/2*
  - *file/1*
  - *save/1*
  - *save/0*
  - *default_option/1*
  - *default_options/1*
- *Protected predicates*
- *Private predicates*
  - *merge_options/2*
  - *predicate_called_but_not_defined_/2*
  - *object_predicate_called_/3*
  - *module_predicate_called_/3*
  - *unknown_predicate_called_/2*
  - *missing_predicate_directive_/3*
  - *non_standard_predicate_call_/2*
  - *dynamic_directive_/3*
  - *multifile_directive_/3*
  - *add_directive_before_entity_/2*
  - *add_directive_/2*
  - *add_directive_/3*
  - *remove_directive_/2*
  - *file_being_advised_/4*
- *Operators*

Public predicates

rdirectory/2

Advises the user on missing directives for converting all plain Prolog files in a directory and its sub-directories to Logtalk objects using the specified options.

Compilation flags:

```
static
```

Template:

```
rdirectory(Directory,Options)
```

Mode and number of proofs:

```
rdirectory(+atom,+list(compound)) - one
```

rdirectory/1

Advises the user on missing directives for converting all plain Prolog files in a directory and its sub-directories to Logtalk objects using default options.

Compilation flags:

```
static
```

Template:

```
rdirectory(Directory)
```

Mode and number of proofs:

```
rdirectory(+atom) - one
```

directory/2

Advises the user on missing directives for converting all plain Prolog files in a directory to Logtalk objects using the specified options.

Compilation flags:

```
static
```

Template:

```
directory(Directory,Options)
```

Mode and number of proofs:

```
directory(+atom,+list(compound)) - one
```

directory/1

Advises the user on missing directives for converting all plain Prolog files in a directory to Logtalk objects using default options.

Compilation flags:

```
static
```

Template:

```
directory(Directory)
```

Mode and number of proofs:

```
directory(+atom) - one
```

directories/2

Advises the user on missing directives for converting all Prolog files in a set of directories to Logtalk objects using the specified options.

Compilation flags:
`static`

Template:
`directories(Directories,Options)`
Mode and number of proofs:
`directories(+list(atom),+list(compound))` - `one`

---

directories/1

Advises the user on missing directives for converting all Prolog files in a set of directories to Logtalk objects using default options.

Compilation flags:
`static`

Template:
`directories(Directories)`
Mode and number of proofs:
`directories(+list(atom))` - `one`

---

files/2

Advises the user on missing directives for converting a list of plain Prolog files to Logtalk objects using the specified options.

Compilation flags:
`static`

Template:
`files(Files,Options)`
Mode and number of proofs:
`files(+list(atom),+list(compound))` - `one`

---

files/1

Advises the user on missing directives for converting a list of plain Prolog files to Logtalk objects using default options.

Compilation flags:
```
static
```

Template:
```
files(Files)
```
Mode and number of proofs:
```
files(+list(atom)) - one
```

---

file/2

Advises the user on missing directives for converting a plain Prolog file to Logtalk objects using the specified options.

Compilation flags:
```
static
```

Template:
```
file(File,Options)
```
Mode and number of proofs:
```
file(+atom,+list(compound)) - one
```

---

file/1

Advises the user on missing directives for converting a plain Prolog file to Logtalk objects using default options.

Compilation flags:
```
static
```

Template:
```
file(File)
```
Mode and number of proofs:
```
file(+atom) - one
```

---

save/1

Saves the generated wrapper objects (plus a loader file per directory) for all advised files using the specified options. The wrapper objects are saved to the same directories that contain the wrapped Prolog files.

Compilation flags:
`static`

Template:
`save(Options)`
Mode and number of proofs:
`save(+list(compound))` - `one`

---

save/0

Saves the generated wrapper objects (plus a loader file per directory) for all advised files using default options. The wrapper objects are saved to the same directories that contain the wrapped Prolog files.

Compilation flags:
`static`

Mode and number of proofs:
`save` - `one`

---

default_option/1

Enumerates by backtracking the default options used when generating the wrapper objects.

Compilation flags:
`static`

Template:
`default_option(DefaultOption)`
Mode and number of proofs:
`default_option(?compound)` - `zero_or_more`

---

default_options/1

Returns a list of the default options used when generating the wrapper objects.

Compilation flags:
```
static
```

Template:
```
default_options(DefaultOptions)
```
Mode and number of proofs:
```
default_options(-list(compound)) - one
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

merge_options/2

Merges the user options with the default options, returning the list of options used when generating the wrapper objects.

Compilation flags:
```
static
```

Template:
```
merge_options(UserOptions,Options)
```
Mode and number of proofs:
```
merge_options(+list(compound),-list(compound)) - one
```

---

predicate_called_but_not_defined_/2

Table of called object predicates that are not locally defined.

Compilation flags:
```
dynamic
```

Template:
```
predicate_called_but_not_defined_(Object,Predicate)
```
Mode and number of proofs:
```
predicate_called_but_not_defined_(?atom,?predicate_indicator) - zero_or_more
```

---

object_predicate_called_/3

Table of called object predicates.

Compilation flags:
`dynamic`

Template:
`object_predicate_called_(Object,Other,Predicate)`
Mode and number of proofs:
`object_predicate_called_(?atom,?atom,?predicate_indicator)` - `zero_or_more`

---

module_predicate_called_/3

Table of called module predicates.

Compilation flags:
`dynamic`

Template:
`module_predicate_called_(Object,Module,Predicate)`
Mode and number of proofs:
`module_predicate_called_(?atom,?atom,?predicate_indicator)` - `zero_or_more`

---

unknown_predicate_called_/2

Table of predicates called but not defined.

Compilation flags:
`dynamic`

Template:
`unknown_predicate_called_(Object,Predicate)`
Mode and number of proofs:
`unknown_predicate_called_(?atom,?predicate_indicator)` - `zero_or_more`

---

missing_predicate_directive_/3

Table of missing predicate directives.

Compilation flags:
```
dynamic
```

Template:
```
missing_predicate_directive_(Object,Directive,Predicate)
```
Mode and number of proofs:
```
missing_predicate_directive_(?atom,?predicate_indicator,?predicate_indicator) -
zero_or_more
```

---

non_standard_predicate_call_/2

Table of called non-standard predicates.

Compilation flags:
```
dynamic
```

Template:
```
non_standard_predicate_call_(Object,Predicate)
```
Mode and number of proofs:
```
non_standard_predicate_call_(?atom,?predicate_indicator) - zero_or_more
```

---

dynamic_directive_/3

Table of declared dynamic predicates.

Compilation flags:
```
dynamic
```

Template:
```
dynamic_directive_(Object,Line,Predicate)
```
Mode and number of proofs:
```
dynamic_directive_(?atom,?integer,?predicate_indicator) - zero_or_more
```

---

multifile_directive_/3

Table of declared multifile predicates.

Compilation flags:
`dynamic`

Template:
`multifile_directive_(Object,Line,Predicate)`
Mode and number of proofs:
`multifile_directive_(?atom,?integer,?predicate_indicator)` - `zero_or_more`

---

add_directive_before_entity_/2

Table of directives to be added before the entity opening directive.

Compilation flags:
`dynamic`

Template:
`add_directive_before_entity_(Object,Directive)`
Mode and number of proofs:
`add_directive_before_entity_(?atom,?predicate_indicator)` - `zero_or_more`

---

add_directive_/2

Table of directives to be added.

Compilation flags:
`dynamic`

Template:
`add_directive_(Object,Directive)`
Mode and number of proofs:
`add_directive_(?atom,?predicate_indicator)` - `zero_or_more`

---

add_directive_/3

Table of directives to be added to complement existing directives.

Compilation flags:
`dynamic`

Template:
`add_directive_(Object,Directive,NewDirective)`
Mode and number of proofs:
`add_directive_(?atom,?predicate_indicator,?predicate_indicator)` - `zero_or_more`

---

remove_directive_/2

Table of directives to be removed.

Compilation flags:
`dynamic`

Template:
`remove_directive_(Object,Directive)`
Mode and number of proofs:
`remove_directive_(?atom,?predicate_indicator)` - `zero_or_more`

---

file_being_advised_/4

Table of files being advised are respective directories and names (basename without extension).

Compilation flags:
`dynamic`

Template:
`file_being_advised_(File,Path,Directory,Name)`
Mode and number of proofs:
`file_being_advised_(?atom,?atom,?atom,?atom)` - `zero_or_more`

Operators

(none)

1.352 xml_parser

object

1.352.1 `xml`

Bi-directional XML parser.

Availability:
`logtalk_load(xml_parser(loader))`

Author: John Fletcher; adapted to Logtalk by Paulo Moura.
Version: 3:9:0
Date: 2025-10-06

Copyright: Copyright (C) 2001-2005 Binding Time Limited, Copyright (C) 2005-2013 John Fletcher
License: This program is offered free of charge, as unsupported source code. You may use it, copy it, distribute it, modify it or sell it without restriction, but entirely at your own risk.

Compilation flags:
`static, context_switching_calls`

Uses:
list
term

Remarks:

- On-line documentation: https://binding-time.co.uk/index.php/Parsing_XML_with_Prolog
- Compliance: This XML parser supports a subset of XML suitable for XML Data and Worldwide Web applications. It is neither as strict nor as comprehensive as the XML 1.0 Specification mandates.
- Compliance-strictness: It is not as strict, because, while the specification must eliminate ambiguities, not all errors need to be regarded as faults, and some reasonable examples of real XML usage would have to be rejected if they were.
- Compliance-comprehensive: It is not as comprehensive, because, where the XML specification makes provision for more or less complete DTDs to be provided as part of a document, xml.pl actions the local definition of ENTITIES only. Other DTD extensions are treated as commentary.
- Bi-directional conversions: Conversions are not fully symmetrical as weaker XML is accepted than can be generated. Notably, in-bound (Codes -> Document) parsing does not require strictly well-formed XML. If Codes does not represent well-formed XML, Document is instantiated to the term `malformed(<attributes>,<content>)`.

Inherited public predicates:
(none)

- *Public predicates*

    - *parse/2*
    - *parse/3*
    - *subterm/2*
    - *pp/1*
- *Protected predicates*
- *Private predicates*
    - *xml_to_document/3*
    - *empty_map/1*
    - *map_member/3*
    - *map_store/4*
    - *pp_string/1*
    - *fault/5*
    - *exception/4*
    - *document_generation//2*
    - *pcdata_7bit//1*
    - *character_data_format/3*
    - *cdata_generation//1*
- *Operators*

Public predicates

parse/2

Parses a list of character codes to/from a data structure of the form `xml(<atts>,<content>)`.

Compilation flags:

```
static
```

Template:

```
parse(Codes,Document)
```

Mode and number of proofs:

```
parse(+list(character_code),?nonvar) - zero_or_one
parse(?list(character_code),+nonvar) - zero_or_one
```

---

parse/3

Parses a list of character codes to/from a data structure of the form `xml(<atts>,<content>)` using the given list of options.

Compilation flags:

```
static
```

Template:

```
parse(Options,Codes,Document)
```

Mode and number of proofs:

```
parse(++list(compound),+list(character_code),?nonvar) - zero_or_one
parse(++list(compound),?list(character_code),+nonvar) - zero_or_one
```

Remarks:

- `extended_characters(Boolean)` option: Use the extended character entities for XHTML (default `true`).
- `format(Boolean)` option: For parsing, strip layouts when no character data appears between elements (default `true`). For generating, indent the element content (default `true`).
- `remove_attribute_prefixes(Boolean)` option: Remove namespace prefixes from attributes when it's the same as the prefix of the parent element (default `false`).
- `allow_ampersand(Boolean)` option: Allow unescaped ampersand characters (&) to occur in PCDATA (default `false`).

---

subterm/2

Unifies `Subterm` with a sub-term of `XMLTerm`. Note that `XMLTerm` is a sub-term of itself.

Compilation flags:
`static`

Template:
`subterm(XMLTerm,Subterm)`
Mode and number of proofs:
`subterm(+nonvar,?nonvar)` - `zero_or_one`

---

pp/1

Pretty prints a XML document on the current output stream.

Compilation flags:
`static`

Template:
`pp(XMLDocument)`
Mode and number of proofs:
`pp(+nonvar)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

xml_to_document/3

Translates the list of character codes `XML` into the Prolog term `Document`. `Options` is a list of terms controlling the treatment of layout characters and character entities.

Compilation flags:

```
static
```

Template:

```
xml_to_document(Options,XML,Document)
```

Mode and number of proofs:

```
xml_to_document(+nonvar,+nonvar,?nonvar) - zero_or_one
```

---

empty_map/1

True if `Map` is a null map.

Compilation flags:

```
static
```

Template:

```
empty_map(Map)
```

Mode and number of proofs:

```
empty_map(?nonvar) - zero_or_one
```

---

map_member/3

True if `Map` is a ordered map structure which records the pair `Key-Data`. `Key` must be ground.

Compilation flags:

```
static
```

Template:

```
map_member(Key,Map,Data)
```

Mode and number of proofs:

```
map_member(+nonvar,+nonvar,?nonvar) - zero_or_one
```

---

map_store/4

True if Map0 is an ordered map structure, Key must be ground, and Map1 is identical to Map0 except that the pair Key-Data is recorded by Map1.

Compilation flags:
static

Template:
map_store(Map0,Key,Data,Map1)

Mode and number of proofs:
map_store(+nonvar,+nonvar,+nonvar,?nonvar) - zero_or_one

---

pp_string/1

Prints String onto the current output stream. If String contains only 7-bit chars it is printed in shorthand quoted format, otherwise it is written as a list.

Compilation flags:
static

Template:
pp_string(String)

Mode and number of proofs:
pp_string(+nonvar) - zero_or_one

---

fault/5

Identifies SubTerm as a sub-term of Term which cannot be serialized after Indentation. Message is an atom naming the type of error; Path is a string encoding a list of SubTerm's ancestor elements in the form <tag>{(id)}* where <tag> is the element tag and <id> is the value of any attribute _named_ id.

Compilation flags:
static

Template:
fault(Term,Indentation,SubTerm,Path,Message)

Mode and number of proofs:
fault(+nonvar,+nonvar,?nonvar,?nonvar,?nonvar) - zero_or_one

---

exception/4

Hook to raise an exception to be raised in respect of a fault in the XML Term `Document`.

Compilation flags:

```
static
```

Template:

```
exception(Message,Document,Culprit,Path)
```

Mode and number of proofs:

```
exception(+atom,+nonvar,+nonvar,+nonvar) - one
```

---

document_generation//2

DCG generating Document as a list of character codes. `Format` is true|false defining whether layouts, to provide indentation, should be added between the element content of the resultant "string". Note that formatting is disabled for elements that are interspersed with `pcdata/1` terms, such as XHTML's 'inline' elements. Also, `Format` is over-ridden, for an individual element, by an explicit `'xml:space'="preserve"` attribute.

Compilation flags:

```
static
```

Template:

```
document_generation(Format,Document)
```

Mode and number of proofs:

```
document_generation(+nonvar,+nonvar) - zero_or_one
```

---

pcdata_7bit//1

Represents the ASCII character set in its simplest format, using the character entities `&`, `"`, `<`, and `>` which are common to both XML and HTML. The numeric entity `'` is used in place of `'` because browsers don't recognize it in HTML.

Compilation flags:

```
static
```

Template:

```
pcdata_7bit(Code)
```

Mode and number of proofs:

```
pcdata_7bit(?nonvar) - zero_or_one
```

---

character_data_format/3

Holds when `Format0` and `Format1` are the statuses of XML formatting before and after `Codes` - which may be null.

Compilation flags:
`static`

Template:
`character_data_format(Codes,Format0,Format1)`
Mode and number of proofs:
`character_data_format(+nonvar,+nonvar,?nonvar)` - `zero_or_one`

---

cdata_generation//1

Holds when `Format0` and `Format1` are the statuses of XML formatting before and after `Codes` - which may be null.

Compilation flags:
`static`

Template:
`cdata_generation(Codes)`
Mode and number of proofs:
`cdata_generation(+list)` - `zero_or_one`

Operators

(none)

1.353 yaml

object

1.353.1 yaml

YAML parser and generator.

Availability:
`logtalk_load(yaml(loader))`

Author: Paulo Moura
Version: 1:1:0
Date: 2026-04-17

Compilation flags:
`static, context_switching_calls`

Implements:
`public` *yaml_protocol*
Uses:
list
reader

Remarks:
(none)

Inherited public predicates:
generate/2 *generate_all/2* *parse/2* *parse_all/2*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)
protocol

1.353.2 `yaml_protocol`

YAML parser and generator protocol.

Availability:
`logtalk_load(yaml(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2026-01-31

Compilation flags:
`static`

Dependencies:

(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *parse/2*
  - *parse_all/2*
  - *generate/2*
  - *generate_all/2*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

parse/2

Parses YAML content read from the given source (`file(Path)`, `stream(Stream)`, `codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) into a ground term representing the parsed YAML data.

Compilation flags:
`static`

Template:
`parse(Source,YAML)`

Mode and number of proofs:
`parse(++compound,--ground)` - `one_or_error`

Exceptions:
`Source` is a variable:
`instantiation_error`
`Source` is neither a variable nor a valid source:
`domain_error(yaml_source,Source)`

parse_all/2

Parses all YAML documents from the given source (`file(Path)`, `stream(Stream)`, `codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) into a list of ground terms. Documents are separated by `---` markers and optionally terminated by `...` markers.

Compilation flags:

`static`

Template:

`parse_all(Source,YAMLs)`

Mode and number of proofs:

`parse_all(++compound,--list(ground))` - `one_or_error`

Exceptions:

`Source` is a variable:

`instantiation_error`

`Source` is neither a variable nor a valid source:

`domain_error(yaml_source,Source)`

---

generate/2

Generates YAML output using the representation specified in the first argument (`file(Path)`, `stream(Stream)`, `codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) from the ground YAML term in the second argument.

Compilation flags:

`static`

Template:

`generate(Sink,YAML)`

Mode and number of proofs:

`generate(++compound,+ground)` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`YAML` is a variable:

`instantiation_error`

`YAML` is not a valid YAML term:

`domain_error(yaml_term,YAML)`

`Sink` cannot be generated:

`domain_error(yaml_sink,Sink)`

---

generate_all/2

Generates YAML output with multiple documents separated by `---` markers using the representation specified in the first argument (`file(Path)`, `stream(Stream)`, `codes(Codes)`, `chars(Chars)`, or `atom(Atom)`) from the list of ground YAML terms in the second argument.

Compilation flags:

`static`

Template:

`generate_all(Sink,YAMLs)`

Mode and number of proofs:

`generate_all(++compound,+list(ground))` - `one_or_error`

Exceptions:

`Sink` is a variable:

`instantiation_error`

`YAMLs` is a variable:

`instantiation_error`

`YAMLs` is not a list:

`type_error(list,YAMLs)`

An element of `YAMLs` is not a valid YAML term:

`domain_error(yaml_term,Term)`

`Sink` cannot be generated:

`domain_error(yaml_sink,Sink)`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.354 z_score_anomaly_detector

object

1.354.1 z_score_anomaly_detector

Statistical Z-score anomaly detector for continuous datasets. Learns per-attribute population mean and standard deviation from baseline training examples selected from a dataset object implementing the `anomaly_dataset_protocol` protocol and returns a detector term that can be used for scoring, prediction, and export.

Availability:

```
logtalk_load(z_score_anomaly_detector(loader))
```

Author: Paulo Moura
Version: 1:0:0
Date: 2026-05-06

Compilation flags:

```
static, context_switching_calls
```

Imports:

`public` *anomaly_detector_common*

Uses:

format
list
numberlist
pairs
population
type

Remarks:

(none)

Inherited public predicates:

anomaly_detector_options/2 *check_anomaly_detector/1* *check_option/1* *check_options/1* *default_option/1* *default_options/1* *diagnostic/2* *diagnostics/2* *export_to_clauses/4* *export_to_file/4* *learn/2* *learn/3* *option/2* *option/3* *predict/3* *predict/4* *print_anomaly_detector/1* *score/3* *score_all/3* *valid_anomaly_detector/1* *valid_option/1* *valid_options/1*

- *Public predicates*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

See also

anomaly_dataset_protocol, *anomaly_detector_protocol*, *modified_z_score_anomaly_detector*, *isolation_forest_anomaly_detector*, *knn_distance_anomaly_detector*, *lof_anomaly_detector*

1.355 zippers

protocol

1.355.1 `zipperp`

Zipper protocol.

Availability:
`logtalk_load(zippers(loader))`

Author: Paulo Moura
Version: 1:0:0
Date: 2019-01-20

Compilation flags:
`static`

Dependencies:
(none)

Remarks:
(none)

Inherited public predicates:
(none)

- *Public predicates*
  - *zip/2*
  - *zip/3*
  - *unzip/2*

      - *current/2*
      - *next/2*
      - *next/3*
      - *previous/2*
      - *previous/3*
      - *rewind/2*
      - *rewind/3*
      - *forward/2*
      - *forward/3*
      - *apply/2*
      - *insert_before/3*
      - *insert_after/3*
      - *replace/3*
      - *delete_and_previous/2*
      - *delete_and_next/2*
      - *delete_and_unzip/2*
      - *delete_all_before/2*
      - *delete_all_before_and_unzip/2*
      - *delete_all_after/2*
      - *delete_all_after_and_unzip/2*
    - *Protected predicates*
    - *Private predicates*
    - *Operators*

Public predicates

zip/2

Adds a zipper to a compound term holding a sequence of elements. Fails if the sequence is empty.

Compilation flags:

```
static
```

Template:

```
zip(Sequence,Zipper)
```

Mode and number of proofs:

```
zip(+sequence,--zipper) - zero_or_one
```

zip/3

Adds a zipper to a compound term holding a sequence of elements. Also returns the first element. Fails if the sequence is empty.

Compilation flags:
`static`

Template:
`zip(Sequence,Zipper,First)`
Mode and number of proofs:
`zip(+sequence,--zipper,--term)` - `zero_or_one`

---

unzip/2

Removes a zipper from a sequence.

Compilation flags:
`static`

Template:
`unzip(Zipper,Sequence)`
Mode and number of proofs:
`unzip(@zipper,--sequence)` - `one`

---

current/2

Current element.

Compilation flags:
`static`

Template:
`current(Zipper,Current)`
Mode and number of proofs:
`current(+zipper,?term)` - `zero_or_one`

---

next/2

Moves to the next element. Fails if already at the last elements.

Compilation flags:
`static`

Template:
`next(Zipper,NewZipper)`
Mode and number of proofs:
`next(+zipper,--zipper)` - `zero_or_one`

---

next/3

Moves to and returns the next element. Fails if already at the last elements.

Compilation flags:
`static`

Template:
`next(Zipper,NewZipper,Next)`
Mode and number of proofs:
`next(+zipper,--zipper,-term)` - `zero_or_one`

---

previous/2

Moves to the previous element. Fails if already at the first elements.

Compilation flags:
`static`

Template:
`previous(Zipper,NewZipper)`
Mode and number of proofs:
`previous(+zipper,--zipper)` - `zero_or_one`

---

previous/3

Moves to and returns the previous element. Fails if already at the first element.

Compilation flags:
`static`

Template:
`previous(Zipper,NewZipper,Previous)`
Mode and number of proofs:
`previous(+zipper,--zipper,-term)` - `zero_or_one`

---

rewind/2

Rewinds the zipper so that the first element becomes the current element.

Compilation flags:
`static`

Template:
`rewind(Zipper,NewZipper)`
Mode and number of proofs:
`rewind(+zipper,--zipper)` - `one`

---

rewind/3

Rewinds the zipper so that the first element becomes the current element. Also returns the first element.

Compilation flags:
`static`

Template:
`rewind(Zipper,NewZipper,First)`
Mode and number of proofs:
`rewind(+zipper,--zipper,?term)` - `zero_or_one`

---

forward/2

Forward the zipper so that the last element becomes the current element.

Compilation flags:
`static`

Template:
`forward(Zipper,NewZipper)`
Mode and number of proofs:
`forward(+zipper,--zipper)` - `one`

---

forward/3

Forward the zipper so that the last element becomes the current element. Also returns the last element.

Compilation flags:
`static`

Template:
`forward(Zipper,NewZipper,Last)`
Mode and number of proofs:
`forward(+zipper,--zipper,?term)` - `zero_or_one`

---

apply/2

Applies a closure to the current element.

Compilation flags:
`static`

Template:
`apply(Closure,Zipper)`
Meta-predicate template:
`apply(1,*)`
Mode and number of proofs:
`apply(+callable,+zipper)` - `zero_or_more`

---

insert_before/3

Inserts an element before the current one.

Compilation flags:
`static`

Template:
`insert_before(Zipper,Element,NewZipper)`
Mode and number of proofs:
`insert_before(+zipper,?term,--zipper)` - `zero_or_one`

---

insert_after/3

Inserts an element after the current one.

Compilation flags:
`static`

Template:
`insert_after(Zipper,Element,NewZipper)`
Mode and number of proofs:
`insert_after(+zipper,?term,--zipper)` - `zero_or_one`

---

replace/3

Replaces the current element with a new element.

Compilation flags:
`static`

Template:
`replace(Zipper,NewCurrent,NewZipper)`
Mode and number of proofs:
`replace(+zipper,?term,--zipper)` - `one`

---

delete_and_previous/2

Deletes the current element and moves to the previous element. Fails if no previous element exists.

Compilation flags:
`static`

Template:
`delete_and_previous(Zipper,NewZipper)`
Mode and number of proofs:
`delete_and_previous(+zipper,--zipper)` - `zero_or_one`

---

delete_and_next/2

Deletes the current element and moves to the next element. Fails if no next element exists.

Compilation flags:
`static`

Template:
`delete_and_next(Zipper,NewZipper)`
Mode and number of proofs:
`delete_and_next(+zipper,--zipper)` - `zero_or_one`

---

delete_and_unzip/2

Deletes the current element and removes the zipper returning the resulting sequence.

Compilation flags:
`static`

Template:
`delete_and_unzip(Zipper,Sequence)`
Mode and number of proofs:
`delete_and_unzip(+zipper,--sequence)` - `one`

---

delete_all_before/2

Deletes all elements before the current element.

Compilation flags:
`static`

Template:
`delete_all_before(Zipper,NewZipper)`
Mode and number of proofs:
`delete_all_before(+zipper,--zipper) - one`

---

delete_all_before_and_unzip/2

Deletes all elements before the current element and removes the zipper returning the resulting sequence.

Compilation flags:
`static`

Template:
`delete_all_before_and_unzip(Zipper,NewZipper)`
Mode and number of proofs:
`delete_all_before_and_unzip(+zipper,--sequence) - one`

---

delete_all_after/2

Deletes all elements after the current element.

Compilation flags:
`static`

Template:
`delete_all_after(Zipper,NewZipper)`
Mode and number of proofs:
`delete_all_after(+zipper,--zipper) - one`

---

delete_all_after_and_unzip/2

Deletes all elements after the current element and removes the zipper returning the resulting sequence.

Compilation flags:
```
static
```

Template:
```
delete_all_after_and_unzip(Zipper,NewZipper)
```
Mode and number of proofs:
```
delete_all_after_and_unzip(+zipper,--sequence) - one
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

See also

zlist

object

1.355.2 zlist

Zipper list predicates. Zippers should be regarded as opaque terms.

Availability:
```
logtalk_load(zippers(loader))
```

Author: Paulo Moura
Version: 1:0:1
Date: 2019-03-12

Compilation flags:
```
static, context_switching_calls
```

Implements:
```
public zipperp
```

Remarks:
(none)

Inherited public predicates:

apply/2 current/2 delete_all_after/2 delete_all_after_and_unzip/2 delete_all_before/2 delete_all_before_and_unzip/2 delete_and_next/2 delete_and_previous/2 delete_and_unzip/2 forward/2 forward/3 insert_after/3 insert_before/3 next/2 next/3 previous/2 previous/3 replace/3 rewind/2 rewind/3 unzip/2 zip/2 zip/3

- *Public predicates*
  - *zip_at_index/4*
- *Protected predicates*
- *Private predicates*
- *Operators*

Public predicates

zip_at_index/4

Adds a zipper to a list opened at the given index and also returns the element at the index. Fails if the list is empty or the index (starting at 1) does not exist.

Compilation flags:

`static`

Template:

`zip_at_index(Index,List,Zipper,Element)`

Mode and number of proofs:

`zip_at_index(+natural,+list,--zipper,--term)` - `zero_or_one`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

CHAPTER
TWO

DIRECTORIES

To load an entity, always load the library that includes it using the goal `logtalk_load(library_name(loader))` instead of using its path. The library loader file ensures that all the required dependencies are also loaded and that any required flags are used. The loading goal can be found in the entity documentation.

2.1 contributions/flags/

- *flags*
- *flags_validator*

2.2 contributions/iso8601/

- *iso8601*

2.3 contributions/pddl_parser/

- *pddl*
- *read_file*

2.4 contributions/verdi_neruda/

- *a_star_interpreter(W)*
- *benchmark_generators*
- *best_first*
- *bfs_interpreter*
- *bup_interpreter*
- *counter*
- *databasep*
- *debug_expansion(Mode)*
- *demodb*
- *dfs_interpreter*
- *flatting*
- *heuristic_expansion(Mode)*
- *iddfs_interpreter(Increment)*
- *interpreterp*
- *magic*
- *magic_expansion(Mode)*
- *rule_expansion(Mode)*
- *shell*
- *shell(Interpreters)*
- *shell_expansion(Mode)*

2.5 contributions/xml_parser/

- *xml*

2.6 core/

- *core_messages*
- *expanding*
- *forwarding*
- *logtalk*
- *monitoring*
- *user*

2.7 library/

- *cloning*
- *counters*
- *streamvars*

2.8 library/adaptive_boosting_classifier/

- *adaptive_boosting_classifier*

2.9 library/agglomerative_clusterer/

- *agglomerative_clusterer*

2.10 library/amqp/

- *amqp*
- *amqp_pool*

2.11 library/anomaly_detection_protocols/

- *anomaly_dataset_protocol*
- *anomaly_detector_common*
- *anomaly_detector_protocol*

2.12 library/ant_colony/

- *ant_colony(Problem)*
- *ant_colony(Problem,Random)*
- *ant_colony_problem_protocol*

2.13 library/application/

- *application_common*
- *application_protocol*

2.14 library/apriori_pattern_miner/

- *apriori_pattern_miner*

2.15 library/arbitrary/

- *arbitrary*

2.16 library/arrangements/

- *arrangements*
- *arrangements_protocol*

2.17 library/asp/

- *asp*
- *asp_expansion*
- *asp_output*
- *asp_problem_protocol*
- *asp_process*
- *asp_program*
- *asp_protocol*
- *asp_symbols*

2.18 library/assignvars/

- *assignvars*
- *assignvarsp*

2.19 library/association_rule_miner/

- *association_rule_miner*
- *association_rule_miner_common*
- *association_rule_miner_protocol*

2.20 library/atms/

- *atms*
- *atms_bitset_environment*
- *atms_environment_protocol*
- *atms_ordered_list_environment*
- *atms_protocol*
- *atms_segmented_bitset_environment*

2.21 library/avro/

- *avro*

2.22 library/base32/

- *base32*

2.23 library/base58/

- *base58*

2.24 library/base64/

- *base64*
- *base64url*
- *base64url_no_padding*

2.25 library/base85/

- *base85*

2.26 library/baseline_forecasting/

- *baseline_forecasting*

2.27 library/bayesian_ridge_regression/

- *bayesian_ridge_regression*

2.28 library/block_ciphers/

- *aes128*
- *aes192*
- *aes256*
- *aes_common(KeySize,Nk,Nr)*
- *block_cipher_modes_common*
- *block_cipher_prepared_key_protocol*
- *block_cipher_protocol*
- *cbc*
- *ctr*
- *ecb*

2.29 library/bm25_recommender/

- *bm25_recommender*

2.30 library/borda_ranker/

- *borda_ranker*

2.31 library/bradley_terry_ranker/

- *bradley_terry_ranker*

2.32 library/bson/

- *bson*
- *bson(StringRepresentation)*

2.33 library/byte_order/

- *byte_order*

2.34 library/c45_classifier/

- *c45_classifier*

2.35 library/cartesian_products/

- *cartesian_products*
- *cartesian_products_protocol*

2.36 library/cbor/

- *cbor*
- *cbor(StringRepresentation)*

2.37 library/ccsds_frames/

- *ccsds_aos_frames(FrameLength,InsertZoneLength,HasOCF,HasFECF)*
- *ccsds_frame_protocol*
- *ccsds_frames*
- *ccsds_frames_types*
- *ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF)*
- *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*

2.38 library/ccsds_link_profiles/

- *ccsds_link_profiles*

2.39 library/ccsds_packet_services/

- *ccsds_packet_services*

2.40 library/ccsds_packetization/

- *ccsds_packetization*

2.41 library/ccsds_packets/

- *ccsds_packets*
- *ccsds_packets(SecondaryHeaderLength)*
- *ccsds_packets_types*

2.42 library/ccsds_tc_services/

- *ccsds_tc_services*

2.43 library/ccsds_time_codes/

- *ccsds_ccs(CalendarVariant,FractionOctets)*
- *ccsds_cds(DaySegmentOctets,SubmillisecondOctets,Epoch)*
- *ccsds_cuc(CoarseOctets,FineOctets,Epoch)*
- *ccsds_time_code_protocol*
- *ccsds_time_codes*
- *ccsds_time_codes_types*

2.44 library/ccsds_time_fields/

- *ccsds_time_fields*

2.45 library/character_sets/

- *character_set*
- *character_set_protocol*
- *iso_8859_1*
- *iso_8859_10*
- *iso_8859_13*
- *iso_8859_14*
- *iso_8859_15*
- *iso_8859_16*
- *iso_8859_2*
- *iso_8859_3*
- *iso_8859_4*
- *iso_8859_9*
- *mapped_single_byte_character_set*
- *single_byte_character_set(MaxCode)*
- *us_ascii*
- *utf_16_character_set(Endian)*
- *utf_16be*
- *utf_16le*
- *utf_32_character_set(Endian)*
- *utf_32be*
- *utf_32le*
- *utf_8*
- *utf_8_character_set*
- *windows_1250*
- *windows_1251*
- *windows_1252*
- *windows_1253*
- *windows_1254*
- *windows_1257*

2.46 library/chi_square_feature_selector/

- *chi_square_feature_selector*

2.47 library/classification_protocols/

- *classifier_common*
- *classifier_protocol*
- *dataset_protocol*
- *probabilistic_classifier_common*
- *probabilistic_classifier_protocol*

2.48 library/clo_span_pattern_miner/

- *clo_span_pattern_miner*

2.49 library/clustering_protocols/

- *clusterer_common*
- *clusterer_protocol*
- *clustering_dataset_protocol*
- *search_indexing*

2.50 library/cmac/

- *cmac*
- *cmac_protocol*

2.51 library/colley_ranker/

- *colley_ranker*

2.52 library/combinations/

- *combinations*
- *combinations_protocol*

2.53 library/command_line_options/

- *command_line_option*
- *command_line_options*

2.54 library/constrained_optimization/

- *augmented_lagrangian(Problem,InnerSolver)*
- *barrier_sub_problem(Problem,OuterIteration,StartPoint,Mu,Rho,Sign)*
- *constrained_optimization_problem_protocol*
- *constrained_optimization_solver*
- *log_barrier(Problem,InnerSolver)*
- *penalty_sub_problem(Problem,OuterIteration,StartPoint,Rho,Sign)*
- *phase1_sub_problem(Problem,StartPoint)*
- *primal_dual_interior_point(Problem)*
- *qp_active_set*
- *qp_solver_protocol*
- *quadratic_penalty(Problem,InnerSolver)*
- *sqp_active_set(Problem)*
- *sub_problem(Problem,OuterIteration,StartPoint,EqLambda,IneqLambda,Rho,Sign)*

2.55 library/copeland_ranker/

- *copeland_ranker*

2.56 library/coroutining/

- *coroutining*

2.57 library/crontab/

- *crontab*
- *crontab(Representation)*

2.58 library/crs_projections/

- *crs_projections*
- *crs_projections_protocol*

2.59 library/crypto/

- *authenticated_channel*
- *crypto*
- *ed25519*
- *x25519*
- *xchacha20_poly1305*

2.60 library/csv/

- *csv*
- *csv(Header,Separator,IgnoreQuotes)*
- *csv(Header,Separator,IgnoreQuotes,Comments)*
- *csv_guess_questions*
- *csv_protocol*

2.61 library/cuid2/

- *cuid2*
- *cuid2(Representation,Size,Alphabet)*
- *cuid2_protocol*

2.62 library/cusum_anomaly_detector/

- *cusum_anomaly_detector*

2.63 library/datalog/

- *datalog*
- *datalog_protocol*

2.64 library/dates/

- *date*
- *datep*
- *time*
- *timep*

2.65 library/dates_tz/

- *dates_tz*
- *dates_tz_protocol*

2.66 library/dbscan_clusterer/

- *dbscan_clusterer*

2.67 library/dependents/

- *observer*
- *subject*

2.68 library/deques/

- *deque*
- *deque_protocol*

2.69 library/derangements/

- *derangements*
- *derangements_protocol*

2.70 library/dictionaries/

- *avltree*
- *bintree*
- *dictionaryp*
- *rbtree*
- *splaytree*
- *two3tree*

2.71 library/dif/

- *dif*

2.72 library/differential_evolution/

- *differential_evolution(Problem)*
- *differential_evolution(Problem,Random)*
- *differential_evolution_problem_protocol*

2.73 library/dimension_reduction_protocols/

- *dimension_reducer_common*
- *dimension_reducer_protocol*
- *dimension_reduction_dataset_protocol*
- *supervised_dimension_reduction_dataset_protocol*
- *target_supervised_dimension_reduction_dataset_protocol*

2.74 library/eclat_pattern_miner/

- *eclat_pattern_miner*

2.75 library/edcg/

- *edcg*

2.76 library/elastic_net_regression/

- *elastic_net_regression*

2.77 library/elo_ranker/

- *elo_ranker*

2.78 library/emeans_clusterer/

- *emeans_clusterer*
- *emeans_clusterer(Random)*

2.79 library/events/

- *after_event_registry*
- *before_event_registry*
- *event_registry*
- *event_registryp*
- *monitor*
- *monitorp*

2.80 library/ewma_anomaly_detector/

- *ewma_anomaly_detector*

2.81 library/expand_library_alias_paths/

- *expand_library_alias_paths*

2.82 library/expecteds/

- *either*
- *expected*
- *expected(Expected)*

2.83 library/exponential_smoothing/

- *exponential_smoothing*
- *exponential_smoothing_common*
- *exponential_smoothing_problem(Method,Series,OptimizationSeries,Frequency,ParameterSpecification,InitializationSpecification)*

2.84 library/feature_selection_protocols/

- *anova_f_score*
- *chi_square_score*
- *chi_square_score(Discretization)*
- *chi_square_yates_score*
- *correlation_score*
- *cramers_v_bias_corrected_score*

- *cramers_v_bias_corrected_score(Discretization)*
- *cramers_v_score*
- *cramers_v_score(Discretization)*
- *feature_dataset_protocol*
- *feature_discretization*
- *feature_redundancy*
- *feature_scoring_common*
- *feature_scoring_protocol*
- *feature_selector_common*
- *feature_selector_protocol*
- *filter_feature_selector_common*
- *fisher_score*
- *mutual_information_score*
- *mutual_information_score(Discretization)*
- *relief_feature_selector_common*
- *symmetrical_uncertainty_score*
- *symmetrical_uncertainty_score(Discretization)*
- *variance_score*

2.85 library/fisher_score_feature_selector/

- *fisher_score_feature_selector*

2.86 library/flow_networks/

- *capacity_scaling*
- *capacity_scaling(Dictionary)*
- *dinic*
- *dinic(Dictionary)*
- *edmonds_karp*
- *edmonds_karp(Dictionary)*
- *flow_network*
- *flow_network(Dictionary)*
- *flow_network_protocol*
- *flow_support(Dictionary)*
- *gomory_hu*
- *gomory_hu(Dictionary)*
- *max_flow_common(Dictionary)*
- *max_flow_protocol*
- *min_cost_flow_protocol*
- *piecewise_costs*
- *piecewise_costs(Dictionary)*
- *successive_shortest_paths*
- *successive_shortest_paths(Dictionary)*
- *time_expanded_network*
- *time_expanded_network(Dictionary)*

2.87 library/format/

- *format*

2.88 library/fp_growth_pattern_miner/

- *fp_growth_pattern_miner*

2.89 library/frequent_pattern_mining_protocols/

- *frequent_pattern_mining_common*
- *transaction_dataset_protocol*

2.90 library/gaussian_mixture_clusterer/

- *gaussian_mixture_clusterer*

2.91 library/gaussian_process_regression/

- *gaussian_process_regression*

2.92 library/genetic_algorithm/

- *genetic_algorithm(Problem)*
- *genetic_algorithm(Problem,Random)*
- *genetic_algorithm_problem_protocol*

2.93 library/genint/

- *genint*
- *genint_core*

2.94 library/gensym/

- *gensym*
- *gensym_core*

2.95 library/geohash/

- *geohash*
- *geohash_protocol*

2.96 library/geojson/

- *geojson*
- *geojson(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *geojson_protocol*

2.97 library/geospatial/

- *geospatial*
- *geospatial_protocol*

2.98 library/git/

- *git*
- *git_protocol*

2.99 library/glicko2_periodic_ranker/

- *glicko2_periodic_ranker*

2.100 library/glicko2_ranker/

- *glicko2_ranker*

2.101 library/gpx/

- *gpx*
- *gpx_protocol*

2.102 library/gradient_boosting_classifier/

- *gradient_boosting_classifier*

2.103 library/gradient_boosting_regression/

- *gradient_boosting_regression*

2.104 library/grammars/

- *blank_grammars(Format)*
- *ip_grammars(Format)*
- *number_grammars(Format)*
- *sequence_grammars*

2.105 library/graphs/

- *directed_graph_common*
- *directed_graph_protocol*
- *graph_common*
- *graph_protocol*
- *graph_types*
- *undirected_graph_common*
- *unweighted_directed_graph*
- *unweighted_directed_graph(Dictionary)*
- *unweighted_graph_common(Dictionary)*
- *unweighted_graph_protocol*
- *unweighted_undirected_graph*
- *unweighted_undirected_graph(Dictionary)*
- *weighted_directed_graph*
- *weighted_directed_graph(Dictionary)*
- *weighted_graph_common(Dictionary)*
- *weighted_graph_protocol*
- *weighted_undirected_graph*
- *weighted_undirected_graph(Dictionary)*

2.106 library/gravatar/

- *gravatar*
- *gravatar_api*

2.107 library/gsp_pattern_miner/

- *gsp_pattern_miner*

2.108 library/hashes/

- *blake2_core(Key,DigestSize)*
- *blake2b*
- *blake2b(Key,DigestSize)*
- *blake2s*
- *blake2s(Key,DigestSize)*
- *crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength)*
- *crc32_reflected(Polynomial)*
- *crc32b*
- *crc32bzip2*
- *crc32c*
- *crc32mpeg2*
- *crc32posix*
- *crc32q*
- *djb2_32*
- *djb2_64*
- *fips202_fixed_hash(RateBytes,Suffix,OutputBytes)*

- *fips202_hash(RateBytes,Suffix,OutputBytes)*
- *fnv1a_32*
- *fnv1a_64*
- *hash_common_32*
- *hash_common_64*
- *hash_digest_protocol*
- *hash_protocol*
- *hash_state_protocol*
- *md5*
- *murmurhash3_x64_128*
- *murmurhash3_x86_128*
- *murmurhash3_x86_32*
- *sdbm_32*
- *sdbm_64*
- *sha1*
- *sha224*
- *sha256*
- *sha384*
- *sha3_224*
- *sha3_256*
- *sha3_384*
- *sha3_512*
- *sha512*
- *sha512_256*
- *shake128(OutputBytes)*
- *shake256(OutputBytes)*
- *siphash_2_4*
- *siphash_2_4(Key)*

2.109 `library/hdbscan_clusterer/`

- *hdbscan_clusterer*

2.110 `library/heaps/`

- *binary_heap(Order)*
- *binary_heap_max*
- *binary_heap_min*
- *heap(Order)*
- *heap_protocol*
- *heapp*
- *maxheap*
- *minheap*
- *pairing_heap(Order)*
- *pairing_heap_max*
- *pairing_heap_min*

2.111 library/hierarchical_clustering/

- *hierarchical_clustering*

2.112 library/hierarchies/

- *class_hierarchy*
- *class_hierarchyp*
- *hierarchyp*
- *proto_hierarchy*
- *proto_hierarchyp*

2.113 library/hmac/

- *hmac*
- *hmac_protocol*

2.114 library/hodge_rank/

- *hodge_rank*

2.115 library/hook_flows/

- *hook_pipeline(Pipeline)*
- *hook_set(Set)*

2.116 library/hook_objects/

- *backend_adapter_hook*
- *default_workflow_hook*
- *grammar_rules_hook*
- *identity_hook*
- *object_wrapper_hook*
- *object_wrapper_hook(Protocol)*
- *object_wrapper_hook(Name,Relations)*
- *print_goal_hook*
- *prolog_module_hook(Module)*
- *suppress_goal_hook*
- *write_to_file_hook(File)*
- *write_to_file_hook(File,Options)*
- *write_to_stream_hook(Stream)*
- *write_to_stream_hook(Stream,Options)*

2.117 library/html/

- *html*
- *html5*
- *xhtml11*

2.118 library/http_authenticate/

- *http_authenticate*
- *http_authenticate_verifier_protocol*
- *http_htpasswd_verifier(Path)*
- *http_router_basic_auth(Verifier,BaseOptions)*
- *http_server_core_basic_handler(Verifier,Handler,ProtectOptions)*

2.119 library/http_client/

- *http_client*

2.120 library/http_client_core/

- *http_client_core*

2.121 library/http_cookies/

- *http_cookies(Representation)*

2.122 library/http_core/

- *http_authentication_helpers*
- *http_body_codec_protocol*
- *http_core*
- *http_docroot_paths*
- *http_form_body_codec*
- *http_handler_protocol*
- *http_json_body_codec*
- *http_json_term_helpers*
- *http_message_helpers*
- *http_octet_stream_body_codec*
- *http_origin_site_helpers*
- *http_request_protocol*
- *http_response_protocol*
- *http_text_body_codec*
- *http_text_helpers*

2.123 library/http_cors/

- *http_cors*

2.124 library/http_digest/

- *http_client_digest_session*
- *http_client_digest_session(HTTPTransport)*
- *http_digest*
- *http_digest_verifier_protocol*
- *http_router_digest_auth(Verifier,ProtectOptions,AuthenticationInfoOptions)*
- *http_server_core_digest_handler(Verifier,Handler,ProtectOptions,AuthenticationInfoOptions)*

2.125 library/http_directory_listing/

- *http_directory_listing*

2.126 library/http_htmx/

- *http_htmx*
- *http_router_htmx*

2.127 library/http_multipart/

- *http_multipart*

2.128 library/http_oauth/

- *http_oauth*
- *http_oauth_exact_scope_checker*
- *http_oauth_helpers*
- *http_oauth_introspection_client*
- *http_oauth_introspection_verifier(Endpoint,Options)*
- *http_oauth_jwt_verifier(KeyOrJWKSet,Options)*
- *http_oauth_metadata*
- *http_oauth_scope_checker_protocol*
- *http_oauth_verifier_protocol*
- *http_router_oauth(Verifier,BaseOptions)*
- *http_server_core_oauth_endpoint_handler(ProtectedResource,MetadataDescriptors,MetadataOptions,Verifier,Handler,ProtectOptions)*
- *http_server_core_oauth_handler(Verifier,Handler,ProtectOptions)*

2.129 library/http_parameters/

- *http_parameters*
- *http_router_parameters*

2.130 library/http_process_transport/

- *http_process_transport*

2.131 library/http_router/

- *http_router*

2.132 library/http_server/

- *http_server*

2.133 library/http_server_core/

- *http_server_core*

2.134 library/http_session/

- *http_client_session*
- *http_client_session(HTTPTransport)*
- *http_cookie_jar*
- *http_router_server_session(Manager)*
- *http_server_core_session*
- *http_server_core_session_handler(Manager,Handler)*

2.135 library/http_socket_transport/

- *http_socket_transport*
- *http_transport_protocol*

2.136 library/http_sse/

- *http_sse*
- *http_sse_accept_handler(Options)*
- *http_sse_service_handler_protocol*

2.137 library/http_static_files/

- *http_static_files*

2.138 library/http_websocket/

- *http_websocket*
- *http_websocket_accept_handler(Options)*

2.139 library/http_websocket_frames/

- *http_websocket_frames*

2.140 library/http_websocket_handshake/

- *http_websocket_handshake*

2.141 library/http_websocket_messages/

- *http_websocket_messages*
- *http_websocket_messages(TextRepresentation)*

2.142 library/http_websocket_service/

- *http_websocket_client_service*
- *http_websocket_client_service(HTTPTransport)*
- *http_websocket_server_service*
- *http_websocket_server_service(HTTPTransport)*
- *http_websocket_service(HTTPTransport,Role,TextRepresentation)*
- *http_websocket_service_handler_protocol*
- *http_websocket_service_loop_options*
- *http_websocket_service_registry*

2.143 library/http_websocket_session/

- *http_websocket_client_session*
- *http_websocket_server_session*
- *http_websocket_session(Role,TextRepresentation)*
- *http_websocket_session_read_options*
- *http_websocket_session_write_options*

2.144 library/ica_projection/

- *ica_projection*

2.145 library/ids/

- *ids*
- *ids(Representation,Bytes)*

2.146 library/ieee_754/

- *ieee_754(Precision,ByteOrder,NaNRepresentation)*
- *ieee_754_common(Precision,ByteOrder)*
- *ieee_754_fields(Precision,ByteOrder)*
- *ieee_754_fields_protocol*
- *ieee_754_protocol*

2.147 library/integer_partitions/

- *integer_partitions*
- *integer_partitions_protocol*

2.148 library/intermittent_demand_forecasting/

- *intermittent_demand_forecasting*

2.149 library/intervals/

- *interval*
- *interval_algebra*
- *interval_algebra_protocol*
- *interval_constraint_network*
- *interval_constraint_network_protocol*
- *interval_protocol*
- *interval_relation_set*
- *interval_relation_set_protocol*

2.150 library/iqr_anomaly_detector/

- *iqr_anomaly_detector*

2.151 library/iso_13616/

- *iso_13616*
- *iso_13616_protocol*
- *iso_13616_registry*

2.152 library/iso_3166/

- *iso_3166*
- *iso_3166_1*
- *iso_3166_2*
- *iso_3166_protocol*

2.153 library/iso_4217/

- *iso_4217*
- *iso_4217_protocol*

2.154 library/iso_639/

- *iso_639*
- *iso_639_1*
- *iso_639_2*
- *iso_639_3*
- *iso_639_5*
- *iso_639_protocol*

2.155 library/iso_9362/

- *iso_9362*
- *iso_9362_protocol*

2.156 library/isolation_forest_anomaly_detector/

- *isolation_forest_anomaly_detector*

2.157 library/jaccard_recommender/

- *jaccard_recommender*

2.158 library/java/

- *java*
- *java(Reference)*
- *java(Reference,ReturnValue)*
- *java_access_protocol*
- *java_hook*
- *java_utils_protocol*

2.159 library/json/

- *json*
- *json(StringRepresentation)*
- *json(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *json_protocol*

2.160 library/json_graph/

- *json_graph*
- *json_graph_data_protocol*
- *json_graph_protocol*

2.161 library/json_ld/

- *json_ld*
- *json_ld(StringRepresentation)*
- *json_ld(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *json_ld_protocol*

2.162 library/json_lines/

- *json_lines*
- *json_lines(StringRepresentation)*
- *json_lines(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *json_lines_protocol*

2.163 library/json_patch/

- *json_patch*
- *json_patch_protocol*

2.164 library/json_path/

- *json_path*
- *json_path(StringRepresentation)*
- *json_path_protocol*

2.165 library/json_pointer/

- *json_pointer*
- *json_pointer(StringRepresentation)*
- *json_pointer_protocol*

2.166 library/json_rpc/

- *json_rpc*

2.167 library/json_schema/

- *json_schema*
- *json_schema(StringRepresentation)*
- *json_schema(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *json_schema_protocol*

2.168 library/jwt/

- *jwt*
- *jwt_claims*
- *jwt_compact*
- *jwt_der*
- *jwt_helpers*
- *jwt_jwa*
- *jwt_jwk*
- *jwt_jwks*
- *jwt_jws*
- *jwt_openssl*

2.169 library/kcenters_clusterer/

- *kcenters_clusterer*

2.170 library/kemeny_young_ranker/

- *kemeny_young_ranker*

2.171 library/kernel_pca_projection/

- *kernel_pca_projection*

2.172 library/kernel_svm_classifier/

- *kernel_svm_classifier*

2.173 library/kmeans_clusterer/

- *kmeans_clusterer*

2.174 library/kmedians_clusterer/

- *kmedians_clusterer*

2.175 library/kmedoids_clusterer/

- *kmedoids_clusterer*

2.176 library/kmodes_clusterer/

- *kmodes_clusterer*

2.177 library/knn_classifier/

- *knn_classifier*

2.178 library/knn_distance_anomaly_detector/

- *knn_distance_anomaly_detector*

2.179 library/knn_forecasting/

- *knn_forecasting*

2.180 library/knn_item_recommender/

- *knn_item_recommender*

2.181 library/knn_regression/

- *knn_regression*

2.182 library/knn_user_recommender/

- *knn_user_recommender*

2.183 library/kprototypes_clusterer/

- *kprototypes_clusterer*

2.184 library/ksuid/

- *ksuid*
- *ksuid(Representation,Alphabet)*
- *ksuid_protocol*

2.185 library/language_detection/

- *language_detection_scoring*
- *language_detection_scripts*
- *language_detection_strategy_protocol*
- *language_detector(Representation,Strategy)*
- *language_detector_protocol*
- *language_profile_common*
- *language_profile_expansion*
- *language_profile_protocol*
- *language_profiles*

2.186 library/language_detection/profiles/

- *af_language_profile*
- *ar_language_profile*
- *bg_language_profile*
- *bn_language_profile*
- *br_language_profile*
- *ca_language_profile*
- *cs_language_profile*
- *da_language_profile*
- *de_language_profile*
- *el_language_profile*
- *en_language_profile*

- *eo_language_profile*
- *es_language_profile*
- *et_language_profile*
- *eu_language_profile*
- *fa_language_profile*
- *fi_language_profile*
- *fr_language_profile*
- *ga_language_profile*
- *gl_language_profile*
- *ha_language_profile*
- *he_language_profile*
- *hi_language_profile*
- *hr_language_profile*
- *hu_language_profile*
- *hy_language_profile*
- *id_language_profile*
- *it_language_profile*
- *ja_language_profile*
- *ko_language_profile*
- *ku_language_profile*
- *la_language_profile*
- *lt_language_profile*
- *lv_language_profile*
- *mr_language_profile*
- *ms_language_profile*
- *nl_language_profile*
- *no_language_profile*
- *pl_language_profile*
- *pt_language_profile*
- *ro_language_profile*
- *ru_language_profile*
- *sk_language_profile*
- *sl_language_profile*
- *so_language_profile*
- *st_language_profile*
- *sv_language_profile*
- *sw_language_profile*
- *th_language_profile*
- *tl_language_profile*
- *tr_language_profile*
- *uk_language_profile*
- *ur_language_profile*
- *vi_language_profile*
- *yo_language_profile*
- *zh_language_profile*
- *zu_language_profile*

2.187 library/language_detection/strategies/

- *hybrid_language_detector*
- *language_detection_tokenizer*
- *ngram_language_detector*
- *script_language_detector*
- *stopword_language_detector*

2.188 library/lasso_feature_selector/

- *lasso_feature_selector*
- *regression_dataset_adapter(Dataset)*
- *regression_examples_adapter(Declarations,Examples)*

2.189 library/lasso_regression/

- *lasso_regression*

2.190 library/lda_classifier/

- *lda_classifier*

2.191 library/lda_projection/

- *lda_projection*

2.192 library/lemmatization/

- *lemmatizer(Representation,Language)*
- *lemmatizer_language_protocol*
- *lemmatizer_protocol*

2.193 library/lemmatization/languages/

- *english_lemmatizer*

2.194 library/linda/

- *linda*
- *linda_client*
- *linda_server*

2.195 library/linear_algebra/

- *linear_algebra*
- *linear_algebra_protocol*

2.196 library/linear_programming/

- *linear_programming_common*
- *linear_programming_protocol*
- *linear_programming_types*
- *milp_branch_and_bound*
- *simplex*

2.197 library/linear_regression/

- *linear_regression*

2.198 library/linear_svm_classifier/

- *linear_svm_classifier*

2.199 library/listing/

- *listing*

2.200 library/local_optimization/

- *barzilai_borwein(Problem)*
- *bfgs(Problem)*
- *conjugate_gradient(Problem)*
- *gradient_descent(Problem)*
- *lbfgs(Problem)*
- *lbfgs_b(Problem)*
- *local_optimization_problem_protocol*
- *local_optimization_solver(Problem)*
- *nelder_mead(Problem)*
- *trust_region_newton_cg(Problem)*

2.201 library/lof_anomaly_detector/

- *lof_anomaly_detector*

2.202 library/logging/

- *logger*
- *logging*
- *loggingp*

2.203 library/logistic_regression_classifier/

- *logistic_regression_classifier*

2.204 library/loops/

- *loop*
- *loopp*

2.205 library/lru_caches/

- *lru_cache*
- *lru_cache_protocol*

2.206 library/massey_ranker/

- *massey_ranker*

2.207 library/mcp_server/

- *mcp_cache_protocol*
- *mcp_completion_protocol*
- *mcp_multiround_protocol*
- *mcp_prompt_protocol*
- *mcp_resource_protocol*
- *mcp_server*
- *mcp_server_2025_06_18_spec*
- *mcp_server_2025_11_25_spec*
- *mcp_server_2026_07_28_spec*
- *mcp_server_application*
- *mcp_server_spec_protocol*
- *mcp_server_stdio_transport*
- *mcp_server_streamable_http_transport*
- *mcp_server_transport_protocol*
- *mcp_streamable_http_handler*
- *mcp_tool_protocol*
- *mcp_ui_protocol*

2.208 library/memcached/

- *memcached*

2.209 library/message_pack/

- *message_pack(StringRepresentation)*

2.210 library/meta/

- *meta*
- *metap*

2.211 library/meta_compiler/

- *meta_compiler*

2.212 library/mime_types/

- *mime_types*

2.213 library/minizinc/

- *minizinc*
- *minizinc_expansion*
- *minizinc_output*
- *minizinc_problem_protocol*
- *minizinc_process*
- *minizinc_program*
- *minizinc_protocol*

2.214 library/mlp_classifier/

- *mlp_classifier*

2.215 library/modified_z_score_anomaly_detector/

- *modified_z_score_anomaly_detector*

2.216 library/mqtt/

- *mqtt*

2.217 library/mrmr_feature_selector/

- *mrmr_feature_selector*

2.218 library/multisets/

- *multisets*
- *multisets_protocol*

2.219 library/multivariate_distributions/

- *multivariate_distributions(Random)*
- *multivariate_distributions_protocol*

2.220 library/mutations/

- *default_atom_mutations*
- *default_compound_mutations*
- *default_float_mutations*
- *default_integer_mutations*
- *default_list_mutations*
- *mutations*
- *mutations_store*

2.221 library/mutual_information_feature_selector/

- *mutual_information_feature_selector*

2.222 library/n_grams/

- *n_grams(Representation)*
- *n_grams_protocol*

2.223 library/naive_bayes_classifier/

- *naive_bayes_classifier*

2.224 library/nanoid/

- *nanoid*
- *nanoid(Representation,Size,Alphabet)*
- *nanoid_protocol*

2.225 library/nearest_centroid_classifier/

- *nearest_centroid_classifier*

2.226 library/nested_dictionaries/

- *navltree*
- *nbintree*
- *nested_dictionary_protocol*
- *nrbtree*

2.227 library/nmea/

- *nmea*
- *nmea_protocol*

2.228 library/nmf_projection/

- *nmf_projection*

2.229 library/numerical_analysis/

- *adaptive_simpson_quadrature(Function)*
- *barycentric_interpolator*
- *bisection_root_finder(Function)*
- *brent_root_finder(Function)*
- *cubic_spline_interpolator*
- *euler_ode_solver(System)*
- *gauss_legendre_quadrature(Function)*
- *interpolator*
- *interpolator_protocol*
- *newton_root_finder(Function)*
- *ode_solver(System)*
- *ode_solver_protocol*
- *ode_system_protocol*
- *piecewise_linear_interpolator*
- *quadrature(Function)*
- *quadrature_protocol*
- *rk45_ode_solver(System)*
- *rk4_ode_solver(System)*
- *root_finder(Function)*
- *root_finder_protocol*
- *secant_root_finder(Function)*
- *univariate_function_protocol*

2.230 library/omt/

- *omt*
- *omt_expansion*
- *omt_problem_protocol*
- *omt_protocol*

2.231 library/one_class_svm_anomaly_detector/

- *one_class_svm_anomaly_detector*

2.232 library/open_ai/

- *open_ai*
- *open_ai_api*
- *open_ai_backend_protocol*
- *open_ai_catalog*
- *open_ai_client*
- *open_ai_event_stream*
- *open_ai_server(Backend)*

2.233 library/open_api/

- *open_api*
- *open_api_provider_protocol*

2.234 library/open_id/

- *open_id*
- *open_id_client*
- *open_id_der*
- *open_id_discovery*
- *open_id_helpers*
- *open_id_jwks*
- *open_id_jwks_cache*
- *open_id_jwt*
- *open_id_logout*
- *open_id_openssl*
- *open_id_pkce*
- *open_id_response*

2.235 library/optics_clusterer/

- *optics_clusterer*

2.236 library/optionals/

- *maybe*
- *optional*
- *optional(Optional)*

2.237 library/options/

- *options*
- *options_protocol*

2.238 library/os/

- *os*
- *os_types*
- *osp*

2.239 library/otp/

- *otp*
- *otp_protocol*

2.240 library/particle_swarm_optimization/

- *particle_swarm_optimization(Problem)*
- *particle_swarm_optimization(Problem,Random)*
- *particle_swarm_optimization_protocol*

2.241 library/partitions/

- *partitions*
- *partitions_protocol*

2.242 library/paseto/

- *paseto*
- *paseto_claims*
- *paseto_claims_helpers*
- *paseto_helpers*
- *paseto_keys*
- *paseto_protocol*
- *paseto_v4*

2.243 library/pattern_mining_protocols/

- *pattern_miner_common*
- *pattern_miner_protocol*

2.244 library/pca_projection/

- *pca_projection*

2.245 library/permutations/

- *permutations*
- *permutations_protocol*

2.246 library/plackett_luce_last_ranker/

- *plackett_luce_last_ranker*

2.247 library/plackett_luce_ranker/

- *plackett_luce_ranker*

2.248 library/pls_projection/

- *pls_projection*

2.249 library/prefix_span_pattern_miner/

- *prefix_span_pattern_miner*

2.250 library/probabilistic_pca_projection/

- *probabilistic_pca_projection*

2.251 library/problog/

- *problog*
- *problog_expansion*
- *problog_output*
- *problog_problem_protocol*
- *problog_process*
- *problog_program*
- *problog_protocol*

2.252 library/process/

- *process*

2.253 library/protobuf/

- *protobuf*

2.254 library/qda_classifier/

- *qda_classifier*

2.255 library/queues/

- *queue*
- *queuep*

2.256 library/random/

- *backend_random*
- *fast_random*
- *fast_random(Algorithm)*
- *pseudo_random_protocol*
- *random*
- *random(Algorithm)*
- *random_protocol*
- *sampling_protocol*

2.257 library/random_forest_classifier/

- *random_forest_classifier*

2.258 library/random_forest_regression/

- *random_forest_regression*

2.259 library/random_projection/

- *random_projection*

2.260 library/rank_centrality/

- *rank_centrality*

2.261 library/ranked_pairs/

- *ranked_pairs*

2.262 library/ranking_protocols/

- *condorcet_victory_common*
- *glicko2_common*
- *grouped_strength_ranker_common*
- *multiplayer_ranking_dataset_protocol*
- *pairwise_measurement_dataset_protocol*
- *pairwise_ranking_dataset_protocol*
- *pairwise_strength_ranker_common*
- *plackett_luce_common*
- *ranker_common*
- *ranker_protocol*
- *ranking_dataset_common*
- *ranking_dataset_protocol*
- *score_ranker_common*
- *score_ranker_model_common*
- *temporal_pairwise_ranking_dataset_protocol*

2.263 library/reader/

- *reader*

2.264 library/recommender_protocols/

- *cosine_similarity*
- *item_content_dataset_protocol*
- *item_content_dataset_validation*
- *jaccard_similarity*
- *msd_similarity*
- *pearson_similarity*
- *rating_dataset_protocol*
- *recommender_common*
- *recommender_protocol*
- *similarity_metric_common*
- *similarity_metric_protocol*
- *spearman_similarity*

2.265 library/recorded_database/

- *recorded_database*
- *recorded_database_core*

2.266 library/redis/

- *redis*

2.267 library/regression_protocols/

- *regression_dataset_protocol*
- *regressor_common*
- *regressor_protocol*

2.268 library/regression_tree/

- *regression_tree*

2.269 library/regularized_bradley_terry_ranker/

- *regularized_bradley_terry_ranker*

2.270 library/relief_feature_selector/

- *relief_feature_selector*

2.271 library/relieff_feature_selector/

- *relieff_feature_selector*

2.272 library/rest/

- *rest*

2.273 library/ridge_regression/

- *ridge_regression*

2.274 library/ripple_down_rules/

- *generalized_ripple_down_rules*
- *multi_classification_ripple_down_rules*
- *ripple_down_rules_common_protocol*
- *ripple_down_rules_protocol*
- *single_classification_ripple_down_rules*

2.275 library/rrelieff_feature_selector/

- *rrelieff_feature_selector*

2.276 library/s3/

- *s3_client*
- *s3_client(HTTPTransport)*
- *s3_endpoints*
- *s3_headers*
- *s3_protocol*
- *s3_session(HTTPTransport)*
- *s3_signing*
- *s3_xml*

2.277 library/sat/

- *sat*
- *sat_expansion*
- *sat_output*
- *sat_problem_protocol*
- *sat_process*
- *sat_program*
- *sat_protocol*

2.278 library/schulze_ranker/

- *schulze_ranker*

2.279 library/sequential_pattern_mining_protocols/

- *sequence_dataset_protocol*
- *sequential_pattern_mining_common*

2.280 library/sets/

- *set*
- *set(Type)*
- *setp*
- *treap_set*

2.281 library/sgd_classifier/

- *sgd_classifier*

2.282 library/simple_temporal_networks/

- *stn*
- *stn_protocol*

2.283 library/simulated_annealing/

- *simulated_annealing(Problem)*
- *simulated_annealing(Problem,Random)*
- *simulated_annealing_protocol*

2.284 library/slope_one_recommender/

- *slope_one_recommender*

2.285 library/smt/

- *smt*
- *smt_expansion*
- *smt_output*
- *smt_problem_protocol*
- *smt_process*
- *smt_program*
- *smt_protocol*

2.286 library/smtp/

- *smtp*
- *smtp_process_transport*
- *smtp_protocol*

2.287 library/snowflakeid/

- *snowflakeid*
- *snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node)*
- *snowflakeid_instagram*
- *snowflakeid_instagram(Representation)*
- *snowflakeid_protocol*
- *snowflakeid_sonyflake*
- *snowflakeid_sonyflake(Representation)*
- *snowflakeid_twitter*
- *snowflakeid_twitter(Representation)*

2.288 library/sockets/

- *socket*

2.289 library/souffle/

- *souffle*
- *souffle_expansion*
- *souffle_output*
- *souffle_problem_protocol*
- *souffle_process*
- *souffle_program*
- *souffle_protocol*

2.290 library/spade_pattern_miner/

- *spade_pattern_miner*

2.291 library/spectral_clusterer/

- *spectral_clusterer*

2.292 library/sqids/

- *sqids*
- *sqids(Representation)*

2.293 library/statistics/

- *population*
- *sample*
- *statistics*
- *statisticsp*

2.294 library/stemming/

- *lovins_stemmer(Representation)*
- *porter_stemmer(Representation)*
- *stemmer_protocol*

2.295 library/stomp/

- *stomp*

2.296 library/stop_words/

- *stop_words(Representation,Language)*
- *stop_words_language_protocol*
- *stop_words_protocol*

2.297 library/stop_words/languages/

- *stopwords_af*
- *stopwords_ar*
- *stopwords_bg*
- *stopwords_bn*
- *stopwords_br*
- *stopwords_ca*
- *stopwords_cs*
- *stopwords_da*
- *stopwords_de*
- *stopwords_el*
- *stopwords_en*
- *stopwords_eo*
- *stopwords_es*
- *stopwords_et*
- *stopwords_eu*
- *stopwords_fa*
- *stopwords_fi*
- *stopwords_fr*
- *stopwords_ga*
- *stopwords_gl*
- *stopwords_ha*
- *stopwords_he*
- *stopwords_hi*
- *stopwords_hr*
- *stopwords_hu*
- *stopwords_hy*
- *stopwords_id*
- *stopwords_it*
- *stopwords_ja*
- *stopwords_ko*

- *stopwords_ku*
- *stopwords_la*
- *stopwords_lt*
- *stopwords_lv*
- *stopwords_mr*
- *stopwords_ms*
- *stopwords_nl*
- *stopwords_no*
- *stopwords_pl*
- *stopwords_pt*
- *stopwords_ro*
- *stopwords_ru*
- *stopwords_sk*
- *stopwords_sl*
- *stopwords_so*
- *stopwords_st*
- *stopwords_sv*
- *stopwords_sw*
- *stopwords_th*
- *stopwords_tl*
- *stopwords_tr*
- *stopwords_uk*
- *stopwords_ur*
- *stopwords_vi*
- *stopwords_yo*
- *stopwords_zh*
- *stopwords_zu*

2.298 library/string_distance/

- *string_distance(Representation)*

2.299 library/strings/

- *string(Representation)*

2.300 library/subsequences/

- *subsequences*
- *subsequences_protocol*

2.301 library/svr_regression/

- *svr_regression*

2.302 library/tabu_search/

- *tabu_search(Problem)*
- *tabu_search(Problem,Random)*
- *tabu_search_problem_protocol*

2.303 library/term_io/

- *term_io*
- *term_io_protocol*

2.304 library/text_normalization/

- *spelling_normalizer_protocol*
- *text_case_folding*
- *text_diacritics*
- *text_entities*
- *text_normalization_profile_protocol*
- *text_normalizer(Representation,Profile)*
- *text_normalizer_protocol*
- *text_unicode*
- *text_whitespace*

2.305 library/text_normalization/data/

- *xml_whatwg_entities*

2.306 library/text_normalization/profiles/

- *default_text_normalization*

2.307 library/text_vectorization/

- *text_vectorizer*
- *text_vectorizer_protocol*

2.308 library/tfidf_recommender/

- *tfidf_recommender*

2.309 library/theta_forecasting/

- *theta_forecasting*

2.310 library/thurstone_mosteller_ranker/

- *thurstone_mosteller_ranker*

2.311 library/time_scales/

- *time_scales*
- *time_scales_data*
- *time_scales_protocol*

2.312 library/time_series_protocols/

- *forecaster_common*
- *forecaster_protocol*
- *time_series_dataset_protocol*

2.313 library/time_series_regression/

- *time_series_regression*

2.314 library/timeout/

- *timeout*

2.315 library/tle_orbits/

- *tle_orbits*
- *tle_orbits_protocol*

2.316 library/tokenization/

- *sentence_splitter_protocol*
- *tokenizer(Representation,Language)*
- *tokenizer_language_protocol*
- *tokenizer_protocol*
- *tokenizer_rules*

2.317 library/tokenization/languages/

- *english_tokenizer*

2.318 library/toml/

- *toml*
- *toml(StringRepresentation)*
- *toml(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *toml_protocol*

2.319 library/toon/

- *toon*
- *toon(StringRepresentation)*
- *toon(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *toon_protocol*

2.320 library/tries/

- *trie(Representation)*

2.321 library/trueskill_ranker/

- *trueskill_ranker*

2.322 library/truncated_svd_projection/

- *truncated_svd_projection*

2.323 library/tsne_projection/

- *tsne_projection*

2.324 library/tsv/

- *tsv*
- *tsv(Header)*
- *tsv(Header,Comments)*
- *tsv_protocol*

2.325 library/typeid/

- *typeid*
- *typeid(Representation)*
- *typeid_protocol*

2.326 library/types/

- *atom*
- *atomic*
- *callable*
- *character*
- *characterp*
- *comparingp*
- *compound*
- *difflist*
- *float*
- *integer*
- *list*
- *list(Type)*
- *listp*
- *natural*
- *number*
- *numberlist*
- *numberlistp*
- *pairs*
- *term*
- *termp*
- *type*
- *varlist*
- *varlistp*

2.327 library/tzif/

- *tzif*
- *tzif_protocol*
- *tzif_zone_ids*

2.328 library/ulid/

- *ulid*
- *ulid(Representation)*
- *ulid_protocol*
- *ulid_types*

2.329 library/umap_projection/

- *umap_projection*

2.330 library/union_find/

- *union_find*
- *union_find_protocol*

2.331 library/univariate_distributions/

- *univariate_distributions(Random)*
- *univariate_distributions_protocol*

2.332 library/uri_template/

- *uri_template(Representation)*

2.333 library/url/

- *url(Representation)*

2.334 library/uuid/

- *uuid*
- *uuid(Representation)*
- *uuid_protocol*

2.335 library/validations/

- *validated*
- *validation*
- *validation(Validation)*

2.336 library/wkt_wkb/

- *wkt_wkb*
- *wkt_wkb_protocol*

2.337 library/yaml/

- *yaml*
- *yaml_protocol*

2.338 library/z_score_anomaly_detector/

- *z_score_anomaly_detector*

2.339 library/zippers/

- *zipperp*
- *zlist*

2.340 ports/fcube/

- *fcube*

2.341 ports/metagol/

- *metagol*
- *metagol_example_protocol*

2.342 ports/toychr/

- *toychrdb*

2.343 tools/assertions/

- *assertions*
- *assertions(Mode)*
- *assertions_messages*

2.344 tools/code_metrics/

- *cc_metric*
- *code_metric*
- *code_metrics*
- *code_metrics_messages*
- *code_metrics_utilities*
- *cogc_metric*
- *coupling_metric*
- *dit_metric*
- *doc_metric*
- *halstead_metric*
- *halstead_metric(Stroud)*

- *lcom_metric*
- *lines_metric*
- *mi_metric*
- *mi_metric(Stroud)*
- *noc_metric*
- *nor_metric*
- *rfc_metric*
- *size_metric*
- *upn_metric*
- *wmc_metric*

2.345 tools/dead_code_scanner/

- *dead_code_scanner*
- *dead_code_scanner_messages*

2.346 tools/debug_messages/

- *debug_messages*

2.347 tools/debugger/

- *debugger*
- *debugger_messages*
- *debuggerp*
- *dump_trace*

2.348 tools/diagrams/

- *caller_diagram*
- *caller_diagram(Format)*
- *cytoscapejs_graph_language*
- *d2_graph_language*
- *diagram(Format)*
- *diagrams*
- *diagrams(Format)*
- *directory_dependency_diagram*
- *directory_dependency_diagram(Format)*
- *directory_diagram(Format)*
- *directory_load_diagram*
- *directory_load_diagram(Format)*
- *dot_graph_language*
- *entity_diagram*
- *entity_diagram(Format)*
- *file_dependency_diagram*
- *file_dependency_diagram(Format)*
- *file_diagram(Format)*

- *file_load_diagram*
- *file_load_diagram(Format)*
- *graph_language_protocol*
- *graph_language_registry*
- *inheritance_diagram*
- *inheritance_diagram(Format)*
- *library_dependency_diagram*
- *library_dependency_diagram(Format)*
- *library_diagram(Format)*
- *library_load_diagram*
- *library_load_diagram(Format)*
- *mermaid_graph_language*
- *modules_diagram_support*
- *plantuml_graph_language*
- *uses_diagram*
- *uses_diagram(Format)*
- *xref_diagram*
- *xref_diagram(Format)*

2.349 tools/doclet/

- *doclet*

2.350 tools/help/

- *help*

2.351 tools/issue_creator/

- *issue_creator*

2.352 tools/lgtdoc/

- *lgtdoc*
- *lgtdoc_messages*
- *lgtdocp*

2.353 tools/lgtunit/

- *automation_report*
- *cobertura_report*
- *coverage_report*
- *ctrf_output*
- *ctrf_report*
- *lcov_report*
- *lgtunit*
- *lgtunit_messages*

- *minimal_output*
- *sonar_report*
- *subunit_v1_output*
- *subunit_v1_report*
- *subunit_v2_output*
- *subunit_v2_report*
- *tap_output*
- *tap_report*
- *xunit_net_v2_output*
- *xunit_net_v2_report*
- *xunit_output*
- *xunit_report*

2.354 tools/linter_reporter/

- *linter_reporter*

2.355 tools/mutation_testing/

- *clause_mutator_protocol*
- *directive_mutator_protocol*
- *mutation_testing*
- *mutation_testing_messages*
- *mutator_common*
- *mutator_protocol*
- *subprocess_coverage_hook*
- *subprocess_mutation_hook*

2.356 tools/mutation_testing/mutators/

- *arithmetic_operator_replacement(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *body_goal_negation(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *clauses_reordering(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *fail_insertion(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *head_arguments_mutation(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *head_arguments_reordering(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *predicate_directive_suppression(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)*
- *relational_operator_inversion(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *relational_operator_replacement(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *scope_directive_replacement(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)*
- *truth_literal_flip(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *uses_directive_resource_deletion(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)*

2.357 tools/packs/

- *pack_protocol*
- *packs*
- *packs_common*
- *packs_messages*
- *packs_specs_hook*
- *registries*
- *registry_loader_hook*
- *registry_protocol*

2.358 tools/ports_profiler/

- *ports_profiler*

2.359 tools/sarif/

- *sarif*

2.360 tools/sbom/

- *sbom*

2.361 tools/tool_diagnostics/

- *tool_diagnostics_common*
- *tool_diagnostics_protocol*

2.362 tools/tutor/

- *tutor*
- *tutor_explanations*

2.363 tools/wrapper/

- *wrapper*

CHAPTER
THREE

ENTITIES

To load an entity, always load the library that includes it using the goal `logtalk_load(library_name(loader))` instead of loading just the entity. The library loader file ensures that all the required dependencies are also loaded and that any required flags are used. The loading goal can be found in the entity documentation.

3.1 Categories

- *amqp_pool*
- *anomaly_detector_common*
- *application_common*
- *arbitrary*
- *assertions_messages*
- *association_rule_miner_common*
- *authenticated_channel*
- *best_first*
- *blake2_core(Key,DigestSize)*
- *block_cipher_modes_common*
- *ccsds_frames_types*
- *ccsds_packets_types*
- *ccsds_time_codes_types*
- *character_set*
- *class_hierarchy*
- *classifier_common*
- *clusterer_common*
- *code_metric*
- *code_metrics_messages*
- *code_metrics_utilities*
- *command_line_option*
- *condorcet_victory_common*
- *constrained_optimization_solver*
- *core_messages*
- *counters*
- *csv_guess_questions*
- *dead_code_scanner_messages*
- *debugger_messages*
- *diagram(Format)*
- *dimension_reducer_common*
- *directed_graph_common*
- *directory_diagram(Format)*
- *ed25519*

- *exponential_smoothing_common*
- *feature_discretization*
- *feature_redundancy*
- *feature_scoring_common*
- *feature_selector_common*
- *file_diagram(Format)*
- *filter_feature_selector_common*
- *flags*
- *flatting*
- *flow_support(Dictionary)*
- *forecaster_common*
- *frequent_pattern_mining_common*
- *genint_core*
- *gensym_core*
- *glicko2_common*
- *graph_common*
- *graph_types*
- *grouped_strength_ranker_common*
- *html*
- *http_authentication_helpers*
- *http_docroot_paths*
- *http_json_term_helpers*
- *http_message_helpers*
- *http_oauth_helpers*
- *http_origin_site_helpers*
- *http_router*
- *http_router_basic_auth(Verifier,BaseOptions)*
- *http_router_digest_auth(Verifier,ProtectOptions,AuthenticationInfoOptions)*
- *http_router_htmx*
- *http_router_oauth(Verifier,BaseOptions)*
- *http_router_parameters*
- *http_router_server_session(Manager)*
- *http_text_helpers*
- *ieee_754_common(Precision,ByteOrder)*
- *interpolator*
- *item_content_dataset_validation*
- *jwt_helpers*
- *language_detection_scoring*
- *language_profile_common*
- *lgtdoc_messages*
- *lgtunit_messages*
- *library_diagram(Format)*
- *linda_client*
- *linda_server*
- *linear_programming_common*
- *linear_programming_types*
- *listing*
- *local_optimization_solver(Problem)*
- *logging*

- *mapped_single_byte_character_set*
- *max_flow_common(Dictionary)*
- *mcp_server_application*
- *monitor*
- *mutation_testing_messages*
- *mutations*
- *mutator_common*
- *observer*
- *ode_solver(System)*
- *open_id_helpers*
- *options*
- *os_types*
- *packs_common*
- *packs_messages*
- *pairwise_strength_ranker_common*
- *paseto_claims_helpers*
- *paseto_helpers*
- *pattern_miner_common*
- *plackett_luce_common*
- *probabilistic_classifier_common*
- *proto_hierarchy*
- *quadrature(Function)*
- *ranker_common*
- *ranking_dataset_common*
- *read_file*
- *recommender_common*
- *recorded_database_core*
- *regressor_common*
- *relief_feature_selector_common*
- *rest*
- *root_finder(Function)*
- *score_ranker_common*
- *score_ranker_model_common*
- *search_indexing*
- *sequential_pattern_mining_common*
- *similarity_metric_common*
- *single_byte_character_set(MaxCode)*
- *statistics*
- *subject*
- *text_case_folding*
- *text_diacritics*
- *text_entities*
- *text_unicode*
- *text_whitespace*
- *tokenizer_rules*
- *tool_diagnostics_common*
- *tutor_explanations*
- *ulid_types*
- *undirected_graph_common*

- *unweighted_graph_common(Dictionary)*
- *utf_16_character_set(Endian)*
- *utf_32_character_set(Endian)*
- *weighted_graph_common(Dictionary)*
- *x25519*
- *xchacha20_poly1305*

3.2 Objects

- *a_star_interpreter(W)*
- *adaptive_boosting_classifier*
- *adaptive_simpson_quadrature(Function)*
- *aes128*
- *aes192*
- *aes256*
- *aes_common(KeySize,Nk,Nr)*
- *af_language_profile*
- *after_event_registry*
- *agglomerative_clusterer*
- *amqp*
- *anova_f_score*
- *ant_colony(Problem)*
- *ant_colony(Problem,Random)*
- *apriori_pattern_miner*
- *ar_language_profile*
- *arithmetic_operator_replacement(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *arrangements*
- *asp*
- *asp_expansion*
- *asp_output*
- *asp_process*
- *asp_program*
- *asp_symbols*
- *assertions*
- *assertions(Mode)*
- *assignvars*
- *association_rule_miner*
- *atms*
- *atms_bitset_environment*
- *atms_ordered_list_environment*
- *atms_segmented_bitset_environment*
- *atom*
- *atomic*
- *augmented_lagrangian(Problem,InnerSolver)*
- *automation_report*
- *avltree*
- *avro*
- *backend_adapter_hook*
- *backend_random*

- *barrier_sub_problem(Problem,OuterIteration,StartPoint,Mu,Rho,Sign)*
- *barycentric_interpolator*
- *barzilai_borwein(Problem)*
- *base32*
- *base58*
- *base64*
- *base64url*
- *base64url_no_padding*
- *base85*
- *baseline_forecasting*
- *bayesian_ridge_regression*
- *before_event_registry*
- *benchmark_generators*
- *bfgs(Problem)*
- *bfs_interpreter*
- *bg_language_profile*
- *binary_heap(Order)*
- *binary_heap_max*
- *binary_heap_min*
- *bintree*
- *bisection_root_finder(Function)*
- *blake2b*
- *blake2b(Key,DigestSize)*
- *blake2s*
- *blake2s(Key,DigestSize)*
- *blank_grammars(Format)*
- *bm25_recommender*
- *bn_language_profile*
- *body_goal_negation(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *borda_ranker*
- *br_language_profile*
- *bradley_terry_ranker*
- *brent_root_finder(Function)*
- *bson*
- *bson(StringRepresentation)*
- *bup_interpreter*
- *byte_order*
- *c45_classifier*
- *ca_language_profile*
- *callable*
- *caller_diagram*
- *caller_diagram(Format)*
- *capacity_scaling*
- *capacity_scaling(Dictionary)*
- *cartesian_products*
- *cbc*
- *cbor*
- *cbor(StringRepresentation)*
- *cc_metric*

- *ccsds_aos_frames(FrameLength,InsertZoneLength,HasOCF,HasFECF)*
- *ccsds_ccs(CalendarVariant,FractionOctets)*
- *ccsds_cds(DaySegmentOctets,SubmillisecondOctets,Epoch)*
- *ccsds_cuc(CoarseOctets,FineOctets,Epoch)*
- *ccsds_frames*
- *ccsds_link_profiles*
- *ccsds_packet_services*
- *ccsds_packetization*
- *ccsds_packets*
- *ccsds_packets(SecondaryHeaderLength)*
- *ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF)*
- *ccsds_tc_services*
- *ccsds_time_codes*
- *ccsds_time_fields*
- *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- *character*
- *chi_square_feature_selector*
- *chi_square_score*
- *chi_square_score(Discretization)*
- *chi_square_yates_score*
- *clauses_reordering(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *clo_span_pattern_miner*
- *cmac*
- *cobertura_report*
- *code_metrics*
- *cogc_metric*
- *colley_ranker*
- *combinations*
- *command_line_options*
- *compound*
- *conjugate_gradient(Problem)*
- *copeland_ranker*
- *coroutining*
- *correlation_score*
- *cosine_similarity*
- *counter*
- *coupling_metric*
- *coverage_report*
- *cramers_v_bias_corrected_score*
- *cramers_v_bias_corrected_score(Discretization)*
- *cramers_v_score*
- *cramers_v_score(Discretization)*
- *crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength)*
- *crc32_reflected(Polynomial)*
- *crc32b*
- *crc32bzip2*
- *crc32c*
- *crc32mpeg2*
- *crc32posix*

- *crc32q*
- *crontab*
- *crontab(Representation)*
- *crs_projections*
- *crypto*
- *cs_language_profile*
- *csv*
- *csv(Header,Separator,IgnoreQuotes)*
- *csv(Header,Separator,IgnoreQuotes,Comments)*
- *ctr*
- *ctrf_output*
- *ctrf_report*
- *cubic_spline_interpolator*
- *cuid2*
- *cuid2(Representation,Size,Alphabet)*
- *cusum_anomaly_detector*
- *cytoscapejs_graph_language*
- *d2_graph_language*
- *da_language_profile*
- *datalog*
- *date*
- *dates_tz*
- *dbscan_clusterer*
- *de_language_profile*
- *dead_code_scanner*
- *debug_expansion(Mode)*
- *debug_messages*
- *debugger*
- *default_atom_mutations*
- *default_compound_mutations*
- *default_float_mutations*
- *default_integer_mutations*
- *default_list_mutations*
- *default_text_normalization*
- *default_workflow_hook*
- *demodb*
- *deque*
- *derangements*
- *dfs_interpreter*
- *diagrams*
- *diagrams(Format)*
- *dif*
- *differential_evolution(Problem)*
- *differential_evolution(Problem,Random)*
- *difflist*
- *dinic*
- *dinic(Dictionary)*
- *directory_dependency_diagram*
- *directory_dependency_diagram(Format)*

- *directory_load_diagram*
- *directory_load_diagram(Format)*
- *dit_metric*
- *djb2_32*
- *djb2_64*
- *doc_metric*
- *doclet*
- *dot_graph_language*
- *dump_trace*
- *ecb*
- *eclat_pattern_miner*
- *edcg*
- *edmonds_karp*
- *edmonds_karp(Dictionary)*
- *either*
- *el_language_profile*
- *elastic_net_regression*
- *elo_ranker*
- *emeans_clusterer*
- *emeans_clusterer(Random)*
- *en_language_profile*
- *english_lemmatizer*
- *english_tokenizer*
- *entity_diagram*
- *entity_diagram(Format)*
- *eo_language_profile*
- *es_language_profile*
- *et_language_profile*
- *eu_language_profile*
- *euler_ode_solver(System)*
- *event_registry*
- *ewma_anomaly_detector*
- *expand_library_alias_paths*
- *expected*
- *expected(Expected)*
- *exponential_smoothing*
- *exponential_smoothing_problem(Method,Series,OptimizationSeries,Frequency,ParameterSpecification,InitializationSpecification)*
- *fa_language_profile*
- *fail_insertion(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *fast_random*
- *fast_random(Algorithm)*
- *fcube*
- *fi_language_profile*
- *file_dependency_diagram*
- *file_dependency_diagram(Format)*
- *file_load_diagram*
- *file_load_diagram(Format)*
- *fips202_fixed_hash(RateBytes,Suffix,OutputBytes)*
- *fips202_hash(RateBytes,Suffix,OutputBytes)*

- *fisher_score*
- *fisher_score_feature_selector*
- *float*
- *flow_network*
- *flow_network(Dictionary)*
- *fnv1a_32*
- *fnv1a_64*
- *format*
- *fp_growth_pattern_miner*
- *fr_language_profile*
- *ga_language_profile*
- *gauss_legendre_quadrature(Function)*
- *gaussian_mixture_clusterer*
- *gaussian_process_regression*
- *generalized_ripple_down_rules*
- *genetic_algorithm(Problem)*
- *genetic_algorithm(Problem,Random)*
- *genint*
- *gensym*
- *geohash*
- *geojson*
- *geojson(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *geospatial*
- *git*
- *gl_language_profile*
- *glicko2_periodic_ranker*
- *glicko2_ranker*
- *gomory_hu*
- *gomory_hu(Dictionary)*
- *gpx*
- *gradient_boosting_classifier*
- *gradient_boosting_regression*
- *gradient_descent(Problem)*
- *grammar_rules_hook*
- *graph_language_registry*
- *gravatar*
- *gravatar_api*
- *gsp_pattern_miner*
- *ha_language_profile*
- *halstead_metric*
- *halstead_metric(Stroud)*
- *hash_common_32*
- *hash_common_64*
- *hdbscan_clusterer*
- *he_language_profile*
- *head_arguments_mutation(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *head_arguments_reordering(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *heap(Order)*
- *help*

- *heuristic_expansion(Mode)*
- *hi_language_profile*
- *hierarchical_clustering*
- *hmac*
- *hodge_rank*
- *hook_pipeline(Pipeline)*
- *hook_set(Set)*
- *hr_language_profile*
- *html5*
- *http_authenticate*
- *http_client*
- *http_client_core*
- *http_client_digest_session*
- *http_client_digest_session(HTTPTransport)*
- *http_client_session*
- *http_client_session(HTTPTransport)*
- *http_cookie_jar*
- *http_cookies(Representation)*
- *http_core*
- *http_cors*
- *http_digest*
- *http_directory_listing*
- *http_form_body_codec*
- *http_htmx*
- *http_htpasswd_verifier(Path)*
- *http_json_body_codec*
- *http_multipart*
- *http_oauth*
- *http_oauth_exact_scope_checker*
- *http_oauth_introspection_client*
- *http_oauth_introspection_verifier(Endpoint,Options)*
- *http_oauth_jwt_verifier(KeyOrJWKSet,Options)*
- *http_oauth_metadata*
- *http_octet_stream_body_codec*
- *http_parameters*
- *http_process_transport*
- *http_server*
- *http_server_core*
- *http_server_core_basic_handler(Verifier,Handler,ProtectOptions)*
- *http_server_core_digest_handler(Verifier,Handler,ProtectOptions,AuthenticationInfoOptions)*
- *http_server_core_oauth_endpoint_handler(ProtectedResource,MetadataDescriptors,MetadataOptions,Verifier,Handler,ProtectOptions)*
- *http_server_core_oauth_handler(Verifier,Handler,ProtectOptions)*
- *http_server_core_session*
- *http_server_core_session_handler(Manager,Handler)*
- *http_socket_transport*
- *http_sse*
- *http_sse_accept_handler(Options)*
- *http_static_files*
- *http_text_body_codec*

- *http_websocket*
- *http_websocket_accept_handler(Options)*
- *http_websocket_client_service*
- *http_websocket_client_service(HTTPTransport)*
- *http_websocket_client_session*
- *http_websocket_frames*
- *http_websocket_handshake*
- *http_websocket_messages*
- *http_websocket_messages(TextRepresentation)*
- *http_websocket_server_service*
- *http_websocket_server_service(HTTPTransport)*
- *http_websocket_server_session*
- *http_websocket_service(HTTPTransport,Role,TextRepresentation)*
- *http_websocket_service_loop_options*
- *http_websocket_service_registry*
- *http_websocket_session(Role,TextRepresentation)*
- *http_websocket_session_read_options*
- *http_websocket_session_write_options*
- *hu_language_profile*
- *hy_language_profile*
- *hybrid_language_detector*
- *ica_projection*
- *id_language_profile*
- *iddfs_interpreter(Increment)*
- *identity_hook*
- *ids*
- *ids(Representation,Bytes)*
- *ieee_754(Precision,ByteOrder,NaNRepresentation)*
- *ieee_754_fields(Precision,ByteOrder)*
- *inheritance_diagram*
- *inheritance_diagram(Format)*
- *integer*
- *integer_partitions*
- *intermittent_demand_forecasting*
- *interval*
- *interval_algebra*
- *interval_constraint_network*
- *interval_relation_set*
- *ip_grammars(Format)*
- *iqr_anomaly_detector*
- *iso8601*
- *iso_13616*
- *iso_13616_registry*
- *iso_3166*
- *iso_3166_1*
- *iso_3166_2*
- *iso_4217*
- *iso_639*
- *iso_639_1*

- *iso_639_2*
- *iso_639_3*
- *iso_639_5*
- *iso_8859_1*
- *iso_8859_10*
- *iso_8859_13*
- *iso_8859_14*
- *iso_8859_15*
- *iso_8859_16*
- *iso_8859_2*
- *iso_8859_3*
- *iso_8859_4*
- *iso_8859_9*
- *iso_9362*
- *isolation_forest_anomaly_detector*
- *issue_creator*
- *it_language_profile*
- *ja_language_profile*
- *jaccard_recommender*
- *jaccard_similarity*
- *java*
- *java(Reference)*
- *java(Reference,ReturnValue)*
- *java_hook*
- *json*
- *json(StringRepresentation)*
- *json(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *json_graph*
- *json_ld*
- *json_ld(StringRepresentation)*
- *json_ld(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *json_lines*
- *json_lines(StringRepresentation)*
- *json_lines(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *json_patch*
- *json_path*
- *json_path(StringRepresentation)*
- *json_pointer*
- *json_pointer(StringRepresentation)*
- *json_rpc*
- *json_schema*
- *json_schema(StringRepresentation)*
- *json_schema(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *jwt*
- *jwt_claims*
- *jwt_compact*
- *jwt_der*
- *jwt_jwa*
- *jwt_jwk*

- *jwt_jwks*
- *jwt_jws*
- *jwt_openssl*
- *kcenters_clusterer*
- *kemeny_young_ranker*
- *kernel_pca_projection*
- *kernel_svm_classifier*
- *kmeans_clusterer*
- *kmedians_clusterer*
- *kmedoids_clusterer*
- *kmodes_clusterer*
- *knn_classifier*
- *knn_distance_anomaly_detector*
- *knn_forecasting*
- *knn_item_recommender*
- *knn_regression*
- *knn_user_recommender*
- *ko_language_profile*
- *kprototypes_clusterer*
- *ksuid*
- *ksuid(Representation,Alphabet)*
- *ku_language_profile*
- *la_language_profile*
- *language_detection_scripts*
- *language_detection_tokenizer*
- *language_detector(Representation,Strategy)*
- *language_profile_expansion*
- *language_profiles*
- *lasso_feature_selector*
- *lasso_regression*
- *lbfgs(Problem)*
- *lbfgs_b(Problem)*
- *lcom_metric*
- *lcov_report*
- *lda_classifier*
- *lda_projection*
- *lemmatizer(Representation,Language)*
- *lgtdoc*
- *lgtunit*
- *library_dependency_diagram*
- *library_dependency_diagram(Format)*
- *library_load_diagram*
- *library_load_diagram(Format)*
- *linda*
- *linear_algebra*
- *linear_regression*
- *linear_svm_classifier*
- *lines_metric*
- *linter_reporter*

- *list*
- *list(Type)*
- *lof_anomaly_detector*
- *log_barrier(Problem,InnerSolver)*
- *logger*
- *logistic_regression_classifier*
- *logtalk*
- *loop*
- *lovins_stemmer(Representation)*
- *lru_cache*
- *lt_language_profile*
- *lv_language_profile*
- *magic*
- *magic_expansion(Mode)*
- *massey_ranker*
- *maxheap*
- *maybe*
- *mcp_server*
- *mcp_server_2025_06_18_spec*
- *mcp_server_2025_11_25_spec*
- *mcp_server_2026_07_28_spec*
- *mcp_server_stdio_transport*
- *mcp_server_streamable_http_transport*
- *mcp_streamable_http_handler*
- *md5*
- *memcached*
- *mermaid_graph_language*
- *message_pack(StringRepresentation)*
- *meta*
- *meta_compiler*
- *metagol*
- *mi_metric*
- *mi_metric(Stroud)*
- *milp_branch_and_bound*
- *mime_types*
- *minheap*
- *minimal_output*
- *minizinc*
- *minizinc_expansion*
- *minizinc_output*
- *minizinc_process*
- *minizinc_program*
- *mlp_classifier*
- *modified_z_score_anomaly_detector*
- *modules_diagram_support*
- *mqtt*
- *mr_language_profile*
- *mrmr_feature_selector*
- *ms_language_profile*

- *msd_similarity*
- *multi_classification_ripple_down_rules*
- *multisets*
- *multivariate_distributions(Random)*
- *murmurhash3_x64_128*
- *murmurhash3_x86_128*
- *murmurhash3_x86_32*
- *mutation_testing*
- *mutations_store*
- *mutual_information_feature_selector*
- *mutual_information_score*
- *mutual_information_score(Discretization)*
- *n_grams(Representation)*
- *naive_bayes_classifier*
- *nanoid*
- *nanoid(Representation,Size,Alphabet)*
- *natural*
- *navltree*
- *nbintree*
- *nearest_centroid_classifier*
- *nelder_mead(Problem)*
- *newton_root_finder(Function)*
- *ngram_language_detector*
- *nl_language_profile*
- *nmea*
- *nmf_projection*
- *no_language_profile*
- *noc_metric*
- *nor_metric*
- *nrbtree*
- *number*
- *number_grammars(Format)*
- *numberlist*
- *object_wrapper_hook*
- *object_wrapper_hook(Protocol)*
- *object_wrapper_hook(Name,Relations)*
- *omt*
- *omt_expansion*
- *one_class_svm_anomaly_detector*
- *open_ai*
- *open_ai_api*
- *open_ai_catalog*
- *open_ai_client*
- *open_ai_event_stream*
- *open_ai_server(Backend)*
- *open_api*
- *open_id*
- *open_id_client*
- *open_id_der*

- *open_id_discovery*
- *open_id_jwks*
- *open_id_jwks_cache*
- *open_id_jwt*
- *open_id_logout*
- *open_id_openssl*
- *open_id_pkce*
- *open_id_response*
- *optics_clusterer*
- *optional*
- *optional(Optional)*
- *os*
- *otp*
- *packs*
- *packs_specs_hook*
- *pairing_heap(Order)*
- *pairing_heap_max*
- *pairing_heap_min*
- *pairs*
- *particle_swarm_optimization(Problem)*
- *particle_swarm_optimization(Problem,Random)*
- *partitions*
- *paseto*
- *paseto_claims*
- *paseto_keys*
- *paseto_v4*
- *pca_projection*
- *pddl*
- *pearson_similarity*
- *penalty_sub_problem(Problem,OuterIteration,StartPoint,Rho,Sign)*
- *permutations*
- *phase1_sub_problem(Problem,StartPoint)*
- *piecewise_costs*
- *piecewise_costs(Dictionary)*
- *piecewise_linear_interpolator*
- *pl_language_profile*
- *plackett_luce_last_ranker*
- *plackett_luce_ranker*
- *plantuml_graph_language*
- *pls_projection*
- *population*
- *porter_stemmer(Representation)*
- *ports_profiler*
- *predicate_directive_suppression(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)*
- *prefix_span_pattern_miner*
- *primal_dual_interior_point(Problem)*
- *print_goal_hook*
- *probabilistic_pca_projection*
- *problog*

- *problog_expansion*
- *problog_output*
- *problog_process*
- *problog_program*
- *process*
- *prolog_module_hook(Module)*
- *protobuf*
- *pt_language_profile*
- *qda_classifier*
- *qp_active_set*
- *quadratic_penalty(Problem,InnerSolver)*
- *queue*
- *random*
- *random(Algorithm)*
- *random_forest_classifier*
- *random_forest_regression*
- *random_projection*
- *rank_centrality*
- *ranked_pairs*
- *rbtree*
- *reader*
- *recorded_database*
- *redis*
- *registries*
- *registry_loader_hook*
- *regression_dataset_adapter(Dataset)*
- *regression_examples_adapter(Declarations,Examples)*
- *regression_tree*
- *regularized_bradley_terry_ranker*
- *relational_operator_inversion(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *relational_operator_replacement(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *relief_feature_selector*
- *relieff_feature_selector*
- *rfc_metric*
- *ridge_regression*
- *rk45_ode_solver(System)*
- *rk4_ode_solver(System)*
- *ro_language_profile*
- *rrelieff_feature_selector*
- *ru_language_profile*
- *rule_expansion(Mode)*
- *s3_client*
- *s3_client(HTTPTransport)*
- *s3_endpoints*
- *s3_headers*
- *s3_session(HTTPTransport)*
- *s3_signing*
- *s3_xml*
- *sample*

- *sarif*
- *sat*
- *sat_expansion*
- *sat_output*
- *sat_process*
- *sat_program*
- *sbom*
- *schulze_ranker*
- *scope_directive_replacement(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)*
- *script_language_detector*
- *sdbm_32*
- *sdbm_64*
- *secant_root_finder(Function)*
- *sequence_grammars*
- *set*
- *set(Type)*
- *sgd_classifier*
- *sha1*
- *sha224*
- *sha256*
- *sha384*
- *sha3_224*
- *sha3_256*
- *sha3_384*
- *sha3_512*
- *sha512*
- *sha512_256*
- *shake128(OutputBytes)*
- *shake256(OutputBytes)*
- *shell*
- *shell(Interpreters)*
- *shell_expansion(Mode)*
- *simplex*
- *simulated_annealing(Problem)*
- *simulated_annealing(Problem,Random)*
- *single_classification_ripple_down_rules*
- *siphash_2_4*
- *siphash_2_4(Key)*
- *size_metric*
- *sk_language_profile*
- *sl_language_profile*
- *slope_one_recommender*
- *smt*
- *smt_expansion*
- *smt_output*
- *smt_process*
- *smt_program*
- *smtp*
- *smtp_process_transport*

- *snowflakeid*
- *snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node)*
- *snowflakeid_instagram*
- *snowflakeid_instagram(Representation)*
- *snowflakeid_sonyflake*
- *snowflakeid_sonyflake(Representation)*
- *snowflakeid_twitter*
- *snowflakeid_twitter(Representation)*
- *so_language_profile*
- *socket*
- *sonar_report*
- *souffle*
- *souffle_expansion*
- *souffle_output*
- *souffle_process*
- *souffle_program*
- *spade_pattern_miner*
- *spearman_similarity*
- *spectral_clusterer*
- *splaytree*
- *sqids*
- *sqids(Representation)*
- *sqp_active_set(Problem)*
- *st_language_profile*
- *stn*
- *stomp*
- *stop_words(Representation,Language)*
- *stopword_language_detector*
- *stopwords_af*
- *stopwords_ar*
- *stopwords_bg*
- *stopwords_bn*
- *stopwords_br*
- *stopwords_ca*
- *stopwords_cs*
- *stopwords_da*
- *stopwords_de*
- *stopwords_el*
- *stopwords_en*
- *stopwords_eo*
- *stopwords_es*
- *stopwords_et*
- *stopwords_eu*
- *stopwords_fa*
- *stopwords_fi*
- *stopwords_fr*
- *stopwords_ga*
- *stopwords_gl*
- *stopwords_ha*

- *stopwords_he*
- *stopwords_hi*
- *stopwords_hr*
- *stopwords_hu*
- *stopwords_hy*
- *stopwords_id*
- *stopwords_it*
- *stopwords_ja*
- *stopwords_ko*
- *stopwords_ku*
- *stopwords_la*
- *stopwords_lt*
- *stopwords_lv*
- *stopwords_mr*
- *stopwords_ms*
- *stopwords_nl*
- *stopwords_no*
- *stopwords_pl*
- *stopwords_pt*
- *stopwords_ro*
- *stopwords_ru*
- *stopwords_sk*
- *stopwords_sl*
- *stopwords_so*
- *stopwords_st*
- *stopwords_sv*
- *stopwords_sw*
- *stopwords_th*
- *stopwords_tl*
- *stopwords_tr*
- *stopwords_uk*
- *stopwords_ur*
- *stopwords_vi*
- *stopwords_yo*
- *stopwords_zh*
- *stopwords_zu*
- *streamvars*
- *string(Representation)*
- *string_distance(Representation)*
- *sub_problem(Problem,OuterIteration,StartPoint,EqLambda,IneqLambda,Rho,Sign)*
- *subprocess_coverage_hook*
- *subprocess_mutation_hook*
- *subsequences*
- *subunit_v1_output*
- *subunit_v1_report*
- *subunit_v2_output*
- *subunit_v2_report*
- *successive_shortest_paths*
- *successive_shortest_paths(Dictionary)*

- *suppress_goal_hook*
- *sv_language_profile*
- *svr_regression*
- *sw_language_profile*
- *symmetrical_uncertainty_score*
- *symmetrical_uncertainty_score(Discretization)*
- *tabu_search(Problem)*
- *tabu_search(Problem,Random)*
- *tap_output*
- *tap_report*
- *term*
- *term_io*
- *text_normalizer(Representation,Profile)*
- *text_vectorizer*
- *tfidf_recommender*
- *th_language_profile*
- *theta_forecasting*
- *thurstone_mosteller_ranker*
- *time*
- *time_expanded_network*
- *time_expanded_network(Dictionary)*
- *time_scales*
- *time_scales_data*
- *time_series_regression*
- *timeout*
- *tl_language_profile*
- *tle_orbits*
- *tokenizer(Representation,Language)*
- *toml*
- *toml(StringRepresentation)*
- *toml(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *toon*
- *toon(StringRepresentation)*
- *toon(ObjectRepresentation,PairRepresentation,StringRepresentation)*
- *toychrdb*
- *tr_language_profile*
- *treap_set*
- *trie(Representation)*
- *trueskill_ranker*
- *truncated_svd_projection*
- *trust_region_newton_cg(Problem)*
- *truth_literal_flip(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)*
- *tsne_projection*
- *tsv*
- *tsv(Header)*
- *tsv(Header,Comments)*
- *tutor*
- *two3tree*
- *type*

- *typeid*
- *typeid(Representation)*
- *tzif*
- *tzif_zone_ids*
- *uk_language_profile*
- *ulid*
- *ulid(Representation)*
- *umap_projection*
- *union_find*
- *univariate_distributions(Random)*
- *unweighted_directed_graph*
- *unweighted_directed_graph(Dictionary)*
- *unweighted_undirected_graph*
- *unweighted_undirected_graph(Dictionary)*
- *upn_metric*
- *ur_language_profile*
- *uri_template(Representation)*
- *url(Representation)*
- *us_ascii*
- *user*
- *uses_diagram*
- *uses_diagram(Format)*
- *uses_directive_resource_deletion(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)*
- *utf_16be*
- *utf_16le*
- *utf_32be*
- *utf_32le*
- *utf_8*
- *utf_8_character_set*
- *uuid*
- *uuid(Representation)*
- *validated*
- *validation*
- *validation(Validation)*
- *variance_score*
- *varlist*
- *vi_language_profile*
- *weighted_directed_graph*
- *weighted_directed_graph(Dictionary)*
- *weighted_undirected_graph*
- *weighted_undirected_graph(Dictionary)*
- *windows_1250*
- *windows_1251*
- *windows_1252*
- *windows_1253*
- *windows_1254*
- *windows_1257*
- *wkt_wkb*
- *wmc_metric*

- *wrapper*
- *write_to_file_hook(File)*
- *write_to_file_hook(File,Options)*
- *write_to_stream_hook(Stream)*
- *write_to_stream_hook(Stream,Options)*
- *xhtml11*
- *xml*
- *xml_whatwg_entities*
- *xref_diagram*
- *xref_diagram(Format)*
- *xunit_net_v2_output*
- *xunit_net_v2_report*
- *xunit_output*
- *xunit_report*
- *yaml*
- *yo_language_profile*
- *z_score_anomaly_detector*
- *zh_language_profile*
- *zlist*
- *zu_language_profile*

3.3 Protocols

- *anomaly_dataset_protocol*
- *anomaly_detector_protocol*
- *ant_colony_problem_protocol*
- *application_protocol*
- *arrangements_protocol*
- *asp_problem_protocol*
- *asp_protocol*
- *assignvarsp*
- *association_rule_miner_protocol*
- *atms_environment_protocol*
- *atms_protocol*
- *block_cipher_prepared_key_protocol*
- *block_cipher_protocol*
- *cartesian_products_protocol*
- *ccsds_frame_protocol*
- *ccsds_time_code_protocol*
- *character_set_protocol*
- *characterp*
- *class_hierarchyp*
- *classifier_protocol*
- *clause_mutator_protocol*
- *cloning*
- *clusterer_protocol*
- *clustering_dataset_protocol*
- *cmac_protocol*
- *combinations_protocol*

- *comparingp*
- *constrained_optimization_problem_protocol*
- *crs_projections_protocol*
- *csv_protocol*
- *cuid2_protocol*
- *databasep*
- *datalog_protocol*
- *dataset_protocol*
- *datep*
- *dates_tz_protocol*
- *debuggerp*
- *deque_protocol*
- *derangements_protocol*
- *dictionaryp*
- *differential_evolution_problem_protocol*
- *dimension_reducer_protocol*
- *dimension_reduction_dataset_protocol*
- *directed_graph_protocol*
- *directive_mutator_protocol*
- *event_registryp*
- *expanding*
- *feature_dataset_protocol*
- *feature_scoring_protocol*
- *feature_selector_protocol*
- *flags_validator*
- *flow_network_protocol*
- *forecaster_protocol*
- *forwarding*
- *genetic_algorithm_problem_protocol*
- *geohash_protocol*
- *geojson_protocol*
- *geospatial_protocol*
- *git_protocol*
- *gpx_protocol*
- *graph_language_protocol*
- *graph_protocol*
- *hash_digest_protocol*
- *hash_protocol*
- *hash_state_protocol*
- *heap_protocol*
- *heapp*
- *hierarchyp*
- *hmac_protocol*
- *http_authenticate_verifier_protocol*
- *http_body_codec_protocol*
- *http_digest_verifier_protocol*
- *http_handler_protocol*
- *http_oauth_scope_checker_protocol*
- *http_oauth_verifier_protocol*

- *http_request_protocol*
- *http_response_protocol*
- *http_sse_service_handler_protocol*
- *http_transport_protocol*
- *http_websocket_service_handler_protocol*
- *ieee_754_fields_protocol*
- *ieee_754_protocol*
- *integer_partitions_protocol*
- *interpolator_protocol*
- *interpreterp*
- *interval_algebra_protocol*
- *interval_constraint_network_protocol*
- *interval_protocol*
- *interval_relation_set_protocol*
- *iso_13616_protocol*
- *iso_3166_protocol*
- *iso_4217_protocol*
- *iso_639_protocol*
- *iso_9362_protocol*
- *item_content_dataset_protocol*
- *java_access_protocol*
- *java_utils_protocol*
- *json_graph_data_protocol*
- *json_graph_protocol*
- *json_ld_protocol*
- *json_lines_protocol*
- *json_patch_protocol*
- *json_path_protocol*
- *json_pointer_protocol*
- *json_protocol*
- *json_schema_protocol*
- *ksuid_protocol*
- *language_detection_strategy_protocol*
- *language_detector_protocol*
- *language_profile_protocol*
- *lemmatizer_language_protocol*
- *lemmatizer_protocol*
- *lgtdocp*
- *linear_algebra_protocol*
- *linear_programming_protocol*
- *listp*
- *local_optimization_problem_protocol*
- *loggingp*
- *loopp*
- *lru_cache_protocol*
- *max_flow_protocol*
- *mcp_cache_protocol*
- *mcp_completion_protocol*
- *mcp_multiround_protocol*

- *mcp_prompt_protocol*
- *mcp_resource_protocol*
- *mcp_server_spec_protocol*
- *mcp_server_transport_protocol*
- *mcp_tool_protocol*
- *mcp_ui_protocol*
- *metagol_example_protocol*
- *metap*
- *min_cost_flow_protocol*
- *minizinc_problem_protocol*
- *minizinc_protocol*
- *monitoring*
- *monitorp*
- *multiplayer_ranking_dataset_protocol*
- *multisets_protocol*
- *multivariate_distributions_protocol*
- *mutator_protocol*
- *n_grams_protocol*
- *nanoid_protocol*
- *nested_dictionary_protocol*
- *nmea_protocol*
- *numberlistp*
- *ode_solver_protocol*
- *ode_system_protocol*
- *omt_problem_protocol*
- *omt_protocol*
- *open_ai_backend_protocol*
- *open_api_provider_protocol*
- *options_protocol*
- *osp*
- *otp_protocol*
- *pack_protocol*
- *pairwise_measurement_dataset_protocol*
- *pairwise_ranking_dataset_protocol*
- *particle_swarm_optimization_protocol*
- *partitions_protocol*
- *paseto_protocol*
- *pattern_miner_protocol*
- *permutations_protocol*
- *probabilistic_classifier_protocol*
- *problog_problem_protocol*
- *problog_protocol*
- *proto_hierarchyp*
- *pseudo_random_protocol*
- *qp_solver_protocol*
- *quadrature_protocol*
- *queuep*
- *random_protocol*
- *ranker_protocol*

- *ranking_dataset_protocol*
- *rating_dataset_protocol*
- *recommender_protocol*
- *registry_protocol*
- *regression_dataset_protocol*
- *regressor_protocol*
- *ripple_down_rules_common_protocol*
- *ripple_down_rules_protocol*
- *root_finder_protocol*
- *s3_protocol*
- *sampling_protocol*
- *sat_problem_protocol*
- *sat_protocol*
- *sentence_splitter_protocol*
- *sequence_dataset_protocol*
- *setp*
- *similarity_metric_protocol*
- *simulated_annealing_protocol*
- *smt_problem_protocol*
- *smt_protocol*
- *smtp_protocol*
- *snowflakeid_protocol*
- *souffle_problem_protocol*
- *souffle_protocol*
- *spelling_normalizer_protocol*
- *statisticsp*
- *stemmer_protocol*
- *stn_protocol*
- *stop_words_language_protocol*
- *stop_words_protocol*
- *subsequences_protocol*
- *supervised_dimension_reduction_dataset_protocol*
- *tabu_search_problem_protocol*
- *target_supervised_dimension_reduction_dataset_protocol*
- *temporal_pairwise_ranking_dataset_protocol*
- *term_io_protocol*
- *termp*
- *text_normalization_profile_protocol*
- *text_normalizer_protocol*
- *text_vectorizer_protocol*
- *time_scales_protocol*
- *time_series_dataset_protocol*
- *timep*
- *tle_orbits_protocol*
- *tokenizer_language_protocol*
- *tokenizer_protocol*
- *toml_protocol*
- *tool_diagnostics_protocol*
- *toon_protocol*

- *transaction_dataset_protocol*
- *tsv_protocol*
- *typeid_protocol*
- *tzif_protocol*
- *ulid_protocol*
- *union_find_protocol*
- *univariate_distributions_protocol*
- *univariate_function_protocol*
- *unweighted_graph_protocol*
- *uuid_protocol*
- *varlistp*
- *weighted_graph_protocol*
- *wkt_wkb_protocol*
- *yaml_protocol*
- *zipperp*

CHAPTER
FOUR

PREDICATES

This index lists all entities *declaring* a given predicate. To load an entity providing the predicate that you want to call, always load the library that includes it using the goal `logtalk_load(library_name(loader))` instead of loading just the entity. The library loader file ensures that all the required dependencies are also loaded and that any required flags are used. The loading goal can be found in the entity documentation.

- `(/)/2` - *help*
- `(//)/2` - *help*
- `(<)/2` - *comparingp*
- `(<=)/2` - *assignvarsp*, *streamvars*
- `(=:=)/2` - *comparingp*
- `(=<)/2` - *comparingp*
- `(=>)/2` - *assignvarsp*, *streamvars*
- `(=\=)/2` - *comparingp*
- `=~= / 2` - *lgtunit*, *number*
- `(>)/2` - *comparingp*
- `(>=)/2` - *comparingp*
- `abbreviation/2` - *tzif_protocol*
- `abbreviation/3` - *tzif_protocol*
- `abbreviation/4` - *tzif_protocol*
- `abort_transaction/3` - *stomp*
- `absolute_file_name/2` - *osp*
- `accept/3` - *http_sse*, *http_websocket*
- `accept/4` - *http_sse*, *http_websocket*
- `accept_sse/3` - *http_server_core*
- `accept_websocket/3` - *http_server_core*
- `ack/3` - *stomp*
- `acquire/1` - *amqp_pool*
- `activate_debug_handler/1` - *logtalk*
- `activate_monitor/0` - *monitorp*
- `active_debug_handler/1` - *logtalk*
- `add/1` - *registries*
- `add/2` - *registries*
- `add/3` - *difflist*, *registries*
- `add/5` - *memcached*
- `add32/3` - *hash_common_32*
- `add32/4` - *hash_common_32*
- `add32/5` - *hash_common_32*
- `add64/3` - *hash_common_64*
- `addDependent/1` - *subject*
- `add_authentication_info/4` - *http_digest*
- `add_constraint/5` - *stn_protocol*

- add_constraint/6 - *stn_protocol*
- add_constraints/3 - *stn_protocol*
- add_constraints/4 - *stn_protocol*
- add_duration/3 - *datep*
- add_edge/4 - *unweighted_graph_protocol*
- add_edge/5 - *weighted_graph_protocol*
- add_edges/3 - *graph_protocol*
- add_matrices/3 - *linear_algebra_protocol*
- add_response_headers/4 - *http_cors*, *http_htmx*
- add_rule/3 - *datalog_protocol*
- add_scaled_outer_product/4 - *linear_algebra_protocol*
- add_scaled_vector/4 - *linear_algebra_protocol*
- add_time_points/3 - *stn_protocol*
- add_type/2 - *mime_types*
- add_type/3 - *mime_types*
- add_vectors/3 - *linear_algebra_protocol*
- add_vertex/3 - *graph_protocol*
- add_vertices/3 - *graph_protocol*
- adjacent/3 - *geohash_protocol*
- adjacent_bits/4 - *geohash_protocol*
- adjacent_int/4 - *geohash_protocol*
- after/2 - *datep*, *interval_protocol*
- after/3 - *monitoring*
- aggregate_resources/2 - *html*
- alias/1 - *character_set_protocol*
- all/0 - *code_metric*, *dead_code_scanner*, *lgtdocp*
- all/1 - *code_metric*, *dead_code_scanner*, *lgtdocp*
- all_files/0 - *diagram(Format)*, *diagrams(Format)*
- all_files/1 - *diagram(Format)*, *diagrams(Format)*
- all_libraries/0 - *diagram(Format)*, *diagrams(Format)*
- all_libraries/1 - *diagram(Format)*, *diagrams(Format)*
- all_pairs_min_paths/2 - *graph_protocol*
- all_pairs_min_predecessors/2 - *graph_protocol*
- all_score/1 - *code_metric*
- allowed_algorithm/2 - *jwt_jwa*
- along_track_distance/4 - *geospatial_protocol*
- alternating_subsequence/2 - *subsequences_protocol*
- alternating_subsequences/2 - *subsequences_protocol*
- ancestor/1 - *hierarchyp*
- ancestors/1 - *hierarchyp*
- and64/3 - *hash_common_64*
- anomaly_detector_options/2 - *anomaly_detector_protocol*
- api_info/1 - *http_router*, *open_api_provider_protocol*
- apid/2 - *ccsds_packets(SecondaryHeaderLength)*
- apis/0 - *help*
- apis/1 - *help*
- append/2 - *listp*
- append/3 - *listp*, *memcached*, *queuep*, *varlistp*
- append/4 - *redis*

- apply/2 - *zipperp*
- apply/3 - *json_patch_protocol*
- apply/4 - *dictionaryp*
- approximately_equal/2 - *lgtunit*, *number*
- approximately_equal/3 - *lgtunit*, *number*
- apr1/3 - *crypto*
- arbitrary/1 - *arbitrary*
- arbitrary/2 - *arbitrary*
- archive/1 - *registry_protocol*
- arcs/2 - *flow_network_protocol*
- arithmetic_mean/2 - *statisticsp*
- arrangement/3 - *arrangements_protocol*
- arrangement/4 - *arrangements_protocol*
- arrangement_index/4 - *arrangements_protocol*
- arrangement_index/5 - *arrangements_protocol*
- arrangements/3 - *arrangements_protocol*
- arrangements/4 - *arrangements_protocol*
- array_list/2 - *java_utils_protocol*
- array_to_list/2 - *java_utils_protocol*
- array_to_terms/2 - *java_utils_protocol*
- array_to_terms/3 - *java_utils_protocol*
- articulation_points/2 - *undirected_graph_common*
- as_curly_bracketed/2 - *dictionaryp*, *nested_dictionary_protocol*
- as_deque/2 - *deque_protocol*
- as_dictionary/2 - *dictionaryp*
- as_difflist/2 - *list*
- as_heap/2 - *heap_protocol*
- as_list/2 - *deque_protocol*, *dictionaryp*, *difflist*, *heap_protocol*, *lru_cache_protocol*, *queuep*, *ripple_down_rules_common_protocol*, *setp*, *trie(Representation)*
- as_list/3 - *trie(Representation)*
- as_nested_dictionary/2 - *nested_dictionary_protocol*
- as_set/2 - *setp*
- as_trie/2 - *trie(Representation)*
- ask_question/5 - *logtalk*
- asp_statement/2 - *asp_problem_protocol*
- aspiration/3 - *tabu_search_problem_protocol*
- assert_fact/1 - *datalog_protocol*
- assertion/1 - *assertions(Mode)*, *lgtunit*
- assertion/2 - *assertions(Mode)*, *lgtunit*
- assignable/1 - *assignvarsp*
- assignable/2 - *assignvarsp*
- association_rule_miner_options/2 - *association_rule_miner_protocol*
- assumption/2 - *atms_protocol*
- assumptions/2 - *atms_protocol*
- atom_string/2 - *string(Representation)*
- atomics_to_string/2 - *string(Representation)*
- atomics_to_string/3 - *string(Representation)*
- attach_sse_stream/1 - *mcp_server_streamable_http_transport*
- attribute_values/2 - *anomaly_dataset_protocol*, *clustering_dataset_protocol*, *dataset_protocol*, *dimension_reduction_dataset_protocol*, *feature_dataset_protocol*, *regression_dataset_protocol*

- `authenticate_request/4` - *http_oauth*
- `authenticated_channel_accept/5` - *authenticated_channel*
- `authenticated_channel_decrypt/5` - *authenticated_channel*
- `authenticated_channel_encrypt/5` - *authenticated_channel*
- `authenticated_channel_finalize/3` - *authenticated_channel*
- `authenticated_channel_initiate/4` - *authenticated_channel*
- `authentication_info/2` - *http_digest*
- `authorization/2` - *http_authenticate*, *http_digest*, *http_oauth*
- `authorization_code/4` - *open_id*, *open_id_response*
- `authorization_response/3` - *open_id*, *open_id_response*
- `authorization_url/5` - *open_id*, *open_id_pkce*
- `authorize_request/3` - *http_oauth*
- `authorize_request/6` - *http_digest*
- `available/0` - *packs*
- `available/1` - *packs*
- `available/2` - *packs*
- `avatar_alt_text/2` - *gravatar*
- `avatar_url/2` - *gravatar*
- `average/2` - *numberlistp*
- `average_deviation/3` - *statisticsp*
- `backward_substitution/3` - *linear_algebra_protocol*
- `basic_ack/3` - *amqp*
- `basic_cancel/3` - *amqp*
- `basic_consume/3` - *amqp*
- `basic_get/3` - *amqp*
- `basic_nack/3` - *amqp*
- `basic_publish/4` - *amqp*
- `basic_qos/2` - *amqp*
- `basic_recover/2` - *amqp*
- `basic_reject/3` - *amqp*
- `basis_vector/3` - *linear_algebra_protocol*
- `bbox_contains/2` - *geospatial_protocol*
- `bbox_contains_polygon/2` - *geospatial_protocol*
- `bbox_expand/3` - *geospatial_protocol*
- `bbox_from_coordinates/2` - *geospatial_protocol*
- `bbox_intersects/2` - *geospatial_protocol*
- `bbox_intersects_polygon/2` - *geospatial_protocol*
- `bbox_intersects_polyline/2` - *geospatial_protocol*
- `bbox_overlaps/2` - *geospatial_protocol*
- `bbox_union/3` - *geospatial_protocol*
- `bcrypt/4` - *crypto*
- `before/2` - *datep*, *interval_protocol*
- `before/3` - *monitoring*
- `begin/0` - *datalog_protocol*
- `begin/3` - *http_server_core_session*
- `begin_transaction/3` - *stomp*
- `bell/2` - *natural*
- `bench_goal/1` - *databasep*
- `benchmark/2` - *lgtunit*

- benchmark/3 - *lgtunit*
- benchmark/4 - *lgtunit*
- benchmark_reified/3 - *lgtunit*
- bernoulli/2 - *sampling_protocol*
- beta/3 - *sampling_protocol*, *univariate_distributions_protocol*
- beta_density/4 - *univariate_distributions_protocol*
- beta_distribution/4 - *univariate_distributions_protocol*
- beta_log_density/4 - *univariate_distributions_protocol*
- beta_quantile/4 - *univariate_distributions_protocol*
- beta_samples/4 - *univariate_distributions_protocol*
- between/3 - *integer*, *random_protocol*
- between/4 - *float*
- bic/5 - *iso_9362_protocol*
- bigrams/2 - *n_grams_protocol*
- binary_body/2 - *rest*
- binomial/3 - *natural*, *sampling_protocol*
- bit//1 - *number_grammars(Format)*
- bits//1 - *number_grammars(Format)*
- blank//0 - *blank_grammars(Format)*
- blanks//0 - *blank_grammars(Format)*
- block_size/1 - *block_cipher_protocol*, *hash_digest_protocol*
- body/1 - *http_request_protocol*, *http_response_protocol*
- body/2 - *http_core*
- body/3 - *http_multipart*
- body_pred/1 - *metagol*
- bounding_box/2 - *geohash_protocol*
- bounding_box/3 - *geospatial_protocol*
- bounding_box_bits/3 - *geohash_protocol*
- bounding_box_int/3 - *geohash_protocol*
- bounds/4 - *stn_protocol*
- branch/2 - *git_protocol*
- breadth_first_order/3 - *graph_protocol*
- bridges/2 - *undirected_graph_common*
- broadcast/2 - *http_websocket_service_registry*
- broadcast_except/3 - *http_websocket_service_registry*
- build/4 - *piecewise_costs(Dictionary)*
- build/6 - *time_expanded_network(Dictionary)*
- built_date/1 - *application_protocol*
- built_in_directive/4 - *help*
- built_in_flag/2 - *flags*
- built_in_method/4 - *help*
- built_in_non_terminal/4 - *help*
- built_in_predicate/4 - *help*
- bypass_flag/2 - *ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF)*
- byte_count/1 - *ieee_754_fields_protocol*, *ieee_754_protocol*
- bytes_hex/2 - *hash_common_32*
- bytes_to_codes/2 - *character_set_protocol*
- bytes_to_integer/3 - *byte_order*
- bytes_to_integer/5 - *byte_order*

- bytes_to_signed_integer/3 - *byte_order*
- bytes_to_signed_integer/5 - *byte_order*
- cache/1 - *tzif_protocol*
- cache_policy/4 - *mcp_cache_protocol*
- cache_source/1 - *tzif_protocol*
- cached_jwks/3 - *open_id*, *open_id_jwks_cache*
- cached_tzif/1 - *tzif_protocol*
- calendar_month/3 - *iso8601*
- call_with_timeout/2 - *timeout*
- call_with_timeout/3 - *timeout*
- can_precede/3 - *stn_protocol*
- can_precede_or_equal/3 - *stn_protocol*
- canonical_bic/2 - *iso_9362_protocol*
- canonical_iban/2 - *iso_13616_protocol*
- canonical_query/2 - *s3_signing*
- canonical_request/6 - *s3_signing*
- canonical_uri/4 - *s3_endpoints*
- canonical_uri/5 - *s3_endpoints*
- capabilities/1 - *mcp_tool_protocol*
- capability/2 - *omt_protocol*, *smt_protocol*
- capacity/2 - *lru_cache_protocol*
- cartesian_product/2 - *cartesian_products_protocol*
- cartesian_product/3 - *cartesian_products_protocol*
- cartesian_tuple/2 - *cartesian_products_protocol*
- cartesian_tuple/3 - *cartesian_products_protocol*
- cartesian_tuple_index/3 - *cartesian_products_protocol*
- cartesian_tuple_index/4 - *cartesian_products_protocol*
- cas/6 - *memcached*
- case_conversion/3 - *text_normalization_profile_protocol*
- case_fold/2 - *text_normalizer_protocol*
- cat/2 - *maybe*
- catalan/2 - *natural*
- caverphone/2 - *string_distance(Representation)*
- caverphone_match/2 - *string_distance(Representation)*
- cell_dimensions/3 - *geohash_protocol*
- center_gram_matrix/4 - *linear_algebra_protocol*
- center_gram_vector/4 - *linear_algebra_protocol*
- central_moment/3 - *statisticsp*
- chacha20_block/4 - *xchacha20_poly1305*
- chacha20_encrypt/5 - *xchacha20_poly1305*
- challenge/2 - *http_authenticate*, *http_digest*, *http_oauth*
- change_directory/1 - *osp*
- changed/0 - *subject*
- changed/1 - *subject*
- channel_close/1 - *amqp*
- channel_close/3 - *amqp*
- channel_open/3 - *amqp*
- character_n_grams/3 - *n_grams_protocol*
- character_n_grams/4 - *n_grams_protocol*

- `chebyshev_distance/3` - *numberlistp*
- `chebyshev_norm/2` - *numberlistp*
- `check/0` - *command_line_option*
- `check/1` - *termp*, *varlistp*
- `check/2` - *type*
- `check/3` - *type*
- `check_anomaly_detector/1` - *anomaly_detector_protocol*
- `check_association_rule_miner/1` - *association_rule_miner_protocol*
- `check_classifier/1` - *classifier_protocol*
- `check_clusterer/1` - *clusterer_protocol*
- `check_conversion/3` - *time_scales_protocol*
- `check_convert/4` - *time_scales_protocol*
- `check_dimension_reducer/1` - *dimension_reducer_protocol*
- `check_forecaster/1` - *forecaster_protocol*
- `check_instant/1` - *time_scales_protocol*
- `check_model/1` - *ripple_down_rules_common_protocol*
- `check_offset/3` - *time_scales_protocol*
- `check_option/1` - *options_protocol*
- `check_options/1` - *options_protocol*
- `check_pattern_miner/1` - *pattern_miner_protocol*
- `check_ranker/1` - *ranker_protocol*
- `check_recommender/1` - *recommender_protocol*
- `check_regressor/1` - *regressor_protocol*
- `check_selector/1` - *feature_selector_protocol*
- `check_vectorizer/1` - *text_vectorizer_protocol*
- `checksum/2` - *nmea_protocol*
- `chi_squared/2` - *sampling_protocol*, *univariate_distributions_protocol*
- `chi_squared_density/3` - *univariate_distributions_protocol*
- `chi_squared_distribution/3` - *univariate_distributions_protocol*
- `chi_squared_log_density/3` - *univariate_distributions_protocol*
- `chi_squared_quantile/3` - *univariate_distributions_protocol*
- `chi_squared_samples/3` - *univariate_distributions_protocol*
- `children/2` - *geohash_protocol*
- `cholesky_decomposition/2` - *linear_algebra_protocol*
- `chr_is/2` - *toychrdb*
- `chr_no_spy/1` - *toychrdb*
- `chr_nospy/0` - *toychrdb*
- `chr_notrace/0` - *toychrdb*
- `chr_option/2` - *toychrdb*
- `chr_spy/1` - *toychrdb*
- `chr_trace/0` - *toychrdb*
- `circular_uniform_cartesian/3` - *sampling_protocol*
- `circular_uniform_polar/3` - *sampling_protocol*
- `claim/3` - *jwt*, *jwt_claims*, *paseto*, *paseto_claims*
- `claims/2` - *jwt*, *jwt_compact*, *paseto*
- `class/1` - *anomaly_dataset_protocol*, *class_hierarchyp*, *dataset_protocol*, *supervised_dimension_reduction_dataset_protocol*
- `class_values/1` - *anomaly_dataset_protocol*, *dataset_protocol*, *supervised_dimension_reduction_dataset_protocol*
- `classes/1` - *class_hierarchyp*
- `classifier_options/2` - *classifier_protocol*

- `classify/2` - *ieee_754_fields_protocol*
- `classify/3` - *ripple_down_rules_common_protocol*
- `classify/4` - *ripple_down_rules_common_protocol*
- `classify_with_context/5` - *ripple_down_rules_protocol*
- `clause/5` - *ports_profiler*
- `clause_location/6` - *ports_profiler*
- `clean/0` - *packs*, *registries*
- `clean/1` - *packs*, *registries*
- `clean/2` - *packs*, *text_normalizer_protocol*
- `clean/3` - *text_normalizer_protocol*
- `cleanup/0` - *mcp_server_spec_protocol*, *mcp_server_transport_protocol*
- `clear/0` - *datalog_protocol*, *open_id_jwks_cache*
- `clear/1` - *http_cookie_jar*, *open_id_jwks_cache*
- `clear/2` - *atms_protocol*, *lru_cache_protocol*
- `clear_cache/0` - *tzif_protocol*
- `clear_dut1_override/0` - *time_scales_data*, *time_scales_protocol*
- `clear_leap_seconds_override/0` - *time_scales_data*, *time_scales_protocol*
- `client_open/4` - *socket*
- `client_open/5` - *socket*
- `clockwise_polygon/2` - *geospatial_protocol*
- `clone/1` - *cloning*, *registry_protocol*
- `clone/3` - *dictionaryp*
- `clone/4` - *dictionaryp*
- `close/1` - *amqp*, *http_client_digest_session(HTTPTransport)*, *http_client_session(HTTPTransport)*, *http_cookie_jar*, *http_server*, *http_server_core_session*, *http_sse*, *http_websocket*, *http_websocket_service_registry*, *s3_session(HTTPTransport)*, *smtp_process_transport*
- `close/2` - *http_sse*, *http_websocket*, *socket*
- `close/3` - *amqp*
- `close_client/1` - *linda_client*
- `close_connection/1` - *http_transport_protocol*
- `close_connection_pool/1` - *http_transport_protocol*
- `close_listener/1` - *http_transport_protocol*
- `close_polygon/2` - *geospatial_protocol*
- `cluster/3` - *clusterer_protocol*
- `cluster_probabilities/3` - *gaussian_mixture_clusterer*
- `clusterer_options/2` - *clusterer_protocol*
- `cnf_statement/2` - *sat_problem_protocol*
- `coarse/2` - *ccsds_cuc(CoarseOctets,FineOctets,Epoch)*
- `code_challenge/2` - *open_id_pkce*
- `code_verifier/2` - *open_id_pkce*
- `codes_to_bytes/2` - *character_set_protocol*
- `coefficient_of_variation/2` - *statisticsp*
- `collapse_flow/4` - *piecewise_costs(Dictionary)*
- `combination/3` - *combinations_protocol*
- `combination/4` - *combinations_protocol*
- `combination_index/4` - *combinations_protocol*
- `combination_index/5` - *combinations_protocol*
- `combinations/3` - *combinations_protocol*
- `combinations/4` - *combinations_protocol*

- command_line_arguments/1 - *osp*
- commit/0 - *datalog_protocol*
- commit_author/2 - *git_protocol*
- commit_date/2 - *git_protocol*
- commit_hash/2 - *git_protocol*
- commit_hash_abbreviated/2 - *git_protocol*
- commit_log/3 - *git_protocol*
- commit_message/2 - *git_protocol*
- commit_transaction/3 - *stomp*
- common_prefix/3 - *geohash_protocol*
- common_subsequence/3 - *subsequences_protocol*
- common_subsequences/3 - *subsequences_protocol*
- compact/3 - *json_ld_protocol*
- compact/4 - *jwt_compact*
- company/2 - *gravatar*
- compare_date_time/3 - *datep*
- compatible_key/2 - *jwt_jwk*
- compile_aux_clauses/1 - *logtalk*
- compile_predicate_heads/4 - *logtalk*
- compile_predicate_indicators/3 - *logtalk*
- complement/2 - *unweighted_graph_protocol*
- completion/2 - *help*
- completion/4 - *mcp_completion_protocol*
- completions/2 - *help*
- compose/3 - *interval_algebra_protocol*, *interval_relation_set_protocol*, *typeid_protocol*, *unweighted_directed_graph(Dictionary)*
- compress/2 - *geohash_protocol*
- confirm_select/1 - *amqp*
- connect/1 - *redis*
- connect/2 - *memcached*
- connect/3 - *memcached*, *mqtt*, *redis*
- connect/4 - *amqp*, *mqtt*, *smtp_protocol*, *stomp*
- connected_components/2 - *unweighted_undirected_graph(Dictionary)*, *weighted_undirected_graph(Dictionary)*
- connection_alive/1 - *amqp*, *mqtt*, *smtp_protocol*, *stomp*
- connection_pool_stats/2 - *http_transport_protocol*
- connection_streams/3 - *http_transport_protocol*
- consensus_ranking/2 - *kemeny_young_ranker*
- consensus_score/2 - *kemeny_young_ranker*
- consistent/1 - *interval_constraint_network_protocol*, *stn_protocol*
- consistent/2 - *atms_protocol*
- console/1 - *redis*
- constraint/5 - *linear_programming_protocol*
- constraints/2 - *interval_constraint_network_protocol*, *stn_protocol*
- constraints/3 - *linear_programming_protocol*
- contains/2 - *interval_protocol*
- contradiction/2 - *atms_protocol*, *interval_constraint_network_protocol*
- contradiction_explanations/2 - *interval_constraint_network_protocol*
- control//0 - *blank_grammars(Format)*
- control_command_flag/2 - *ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF)*

- control_construct/4 - *help*
- controls//0 - *blank_grammars(Format)*
- converse/2 - *interval_algebra_protocol*, *interval_relation_set_protocol*
- convert/4 - *time_scales_protocol*
- convert_zones/4 - *dates_tz_protocol*
- convert_zones_with_resolution/5 - *dates_tz_protocol*
- cookie_attribute_present/2 - *http_cookies(Representation)*
- cookie_attribute_value/3 - *http_cookies(Representation)*
- cookie_attribute_value/4 - *http_cookies(Representation)*
- cookie_count/2 - *http_cookie_jar*
- cookie_deletion/3 - *http_cookies(Representation)*
- cookie_expiry/2 - *http_cookies(Representation)*
- cookie_expiry/3 - *http_cookies(Representation)*
- cookie_jar/2 - *http_client_digest_session(HTTPTransport)*, *http_client_session(HTTPTransport)*
- cookies/2 - *http_cookie_jar*
- cooling_schedule/3 - *simulated_annealing_protocol*
- coordinates_bounding_box/2 - *geospatial_protocol*
- copy_file/2 - *osp*
- copy_object/5 - *s3_protocol*
- copy_object/6 - *s3_session(HTTPTransport)*
- copy_object_response/2 - *s3_xml*
- correct_spelling_codes/3 - *spelling_normalizer_protocol*
- correlation/3 - *statisticsp*
- cosine_similarity/3 - *string_distance(Representation)*
- count/2 - *http_server_core_session*, *n_grams_protocol*
- count/3 - *n_grams_protocol*
- count_arrangements/3 - *arrangements_protocol*
- count_cartesian_product/2 - *cartesian_products_protocol*
- count_combinations/3 - *combinations_protocol*
- count_derangements/2 - *derangements_protocol*
- count_distinct_arrangements/3 - *arrangements_protocol*
- count_distinct_cartesian_product/2 - *cartesian_products_protocol*
- count_distinct_combinations/3 - *combinations_protocol*
- count_distinct_derangements/2 - *derangements_protocol*
- count_distinct_multisets/3 - *multisets_protocol*
- count_distinct_partitions/2 - *integer_partitions_protocol*, *partitions_protocol*
- count_distinct_partitions/3 - *integer_partitions_protocol*, *partitions_protocol*
- count_distinct_permutations/2 - *permutations_protocol*
- count_distinct_subsequences/3 - *subsequences_protocol*
- count_multisets/3 - *multisets_protocol*
- count_partial_derangements/3 - *derangements_protocol*
- count_partitions/2 - *integer_partitions_protocol*, *partitions_protocol*
- count_partitions/3 - *integer_partitions_protocol*, *partitions_protocol*
- count_permutations/2 - *permutations_protocol*
- count_subsequences/2 - *subsequences_protocol*
- counter/2 - *counters*, *mutations_store*
- counterclockwise_polygon/2 - *geospatial_protocol*
- country/4 - *iso_3166_1*, *iso_3166_protocol*
- country_spec/3 - *iso_13616_registry*

- `covariance/3` - *statisticsp*
- `covariance_matrix/2` - *linear_algebra_protocol*
- `cover/1` - *lgtunit*
- `coverage_clause_mutator/0` - *mutator_protocol*
- `covering/3` - *geohash_protocol*
- `covering/4` - *geohash_protocol*
- `cpu_time/1` - *osp*, *timep*
- `create/3` - *process*
- `create_assumption/4` - *atms_protocol*
- `create_contradiction/4` - *atms_protocol*
- `create_multipart_upload_response/2` - *s3_xml*
- `create_node/4` - *atms_protocol*
- `created_response/4` - *rest*
- `creators/1` - *application_protocol*
- `cross_track_distance/4` - *geospatial_protocol*
- `crossover/4` - *genetic_algorithm_problem_protocol*
- `crossover_rate/4` - *genetic_algorithm_problem_protocol*
- `crs_dimensions/2` - *crs_projections_protocol*
- `crs_epsg/2` - *crs_projections_protocol*
- `crs_kind/2` - *crs_projections_protocol*
- `crs_name/2` - *crs_projections_protocol*
- `crs_units/2` - *crs_projections_protocol*
- `crypt/5` - *ctr*
- `crypt/6` - *ctr*
- `currency/5` - *iso_4217*, *iso_4217_protocol*
- `current/2` - *http_server_core_session*, *zipperp*
- `current_host/1` - *socket*
- `current_options/1` - *mcp_server_2025_06_18_spec*, *mcp_server_streamable_http_transport*
- `current_url/2` - *http_htmx*
- `current_url_abs_path/2` - *http_htmx*
- `custom_language_scripts/2` - *language_detection_scripts*
- `custom_profile/2` - *language_profiles*
- `cut/3` - *hierarchical_clustering*
- `cut_capacity/3` - *flow_network_protocol*
- `cycle/2` - *directed_graph_protocol*, *undirected_graph_common*
- `damerau_levenshtein/3` - *string_distance(Representation)*
- `data/0` - *ports_profiler*
- `data/1` - *ports_profiler*
- `data/2` - *http_server_core_session*, *nmea_protocol*, *ports_profiler*
- `data_field/2` - *ccsds_frame_protocol*, *ccsds_frames*
- `data_length/2` - *ccsds_packets(SecondaryHeaderLength)*
- `date/4` - *iso8601*
- `date/5` - *iso8601*
- `date/6` - *iso8601*
- `date/7` - *iso8601*
- `date_string/3` - *iso8601*
- `date_time/7` - *osp*
- `date_time_string/3` - *iso8601*
- `date_time_to_unix/2` - *datep*

- day/2 - *ccsds_ccs(CalendarVariant,FractionOctets)*
- day_of_year/2 - *ccsds_ccs(CalendarVariant,FractionOctets)*, *datep*
- day_of_year_date/3 - *datep*
- daylight_saving_time/2 - *tzif_protocol*
- daylight_saving_time/3 - *tzif_protocol*
- daylight_saving_time/4 - *tzif_protocol*
- days/2 - *ccsds_cds(DaySegmentOctets,SubmillisecondOctets,Epoch)*
- days_in_month/3 - *datep*
- deactivate_debug_handler/0 - *logtalk*
- debug/0 - *debuggerp*
- debug_handler/1 - *logtalk*
- debug_handler/3 - *logtalk*
- debugging/0 - *debuggerp*
- debugging/1 - *debuggerp*
- decide/1 - *fcube*
- decide/2 - *fcube*
- decode/2 - *geohash_protocol*, *json_rpc*, *sqids(Representation)*
- decode/3 - *jwt*, *jwt_compact*, *open_id_jwt*, *sqids(Representation)*
- decode/5 - *jwt*, *jwt_compact*, *minizinc_output*
- decode/6 - *problog_output*
- decode/7 - *smt_output*
- decode/8 - *asp_output*, *sat_output*, *souffle_output*
- decode_bits/3 - *geohash_protocol*
- decode_body/4 - *http_body_codec_protocol*, *http_core*
- decode_entities/2 - *text_normalizer_protocol*
- decode_entities/3 - *text_normalizer_protocol*
- decode_exception/2 - *java_utils_protocol*
- decode_exception/3 - *java_utils_protocol*
- decode_frame/2 - *amqp*
- decode_int/3 - *geohash_protocol*
- decode_packet/2 - *mqtt*
- decompile_predicate_heads/4 - *logtalk*
- decompile_predicate_indicators/4 - *logtalk*
- decompose/3 - *typeid_protocol*
- decompose_file_name/3 - *osp*
- decompose_file_name/4 - *osp*
- decompose_flow/3 - *flow_network_protocol*
- decr/3 - *redis*
- decr/4 - *memcached*
- decrby/4 - *redis*
- decrement_counter/1 - *counters*
- decrypt/4 - *ecb*, *paseto*
- decrypt/5 - *cbc*, *paseto*
- decrypt_block/3 - *block_cipher_protocol*
- decrypt_padded/4 - *ecb*
- decrypt_padded/5 - *cbc*
- decrypt_prepared_block/3 - *block_cipher_prepared_key_protocol*
- default/1 - *command_line_option*
- default_language_scripts/2 - *language_detection_scripts*

- `default_option/1` - *options_protocol*, *wrapper*
- `default_options/1` - *options_protocol*, *wrapper*
- `default_profile/2` - *language_profiles*
- `define_log_file/2` - *loggingp*
- `defined/4` - *registries*
- `defined_flag/6` - *flags*
- `degree/3` - *unweighted_undirected_graph(Dictionary)*, *weighted_undirected_graph(Dictionary)*
- `del/3` - *redis*
- `del_monitors/0` - *event_registryp*
- `del_monitors/4` - *event_registryp*
- `del_spy_points/4` - *monitorp*
- `delete/0` - *registries*
- `delete/1` - *registries*
- `delete/2` - *memcached*, *registries*
- `delete/3` - *http_client*, *listp*, *setp*, *varlistp*
- `delete/4` - *dictionaryp*, *heap_protocol*, *http_client*, *http_client_digest_session(HTTPTransport)*, *http_client_session(HTTPTransport)*, *lru_cache_protocol*, *trie(Representation)*
- `delete_all_after/2` - *zipperp*
- `delete_all_after_and_unzip/2` - *zipperp*
- `delete_all_before/2` - *zipperp*
- `delete_all_before_and_unzip/2` - *zipperp*
- `delete_and_next/2` - *zipperp*
- `delete_and_previous/2` - *zipperp*
- `delete_and_unzip/2` - *zipperp*
- `delete_directory/1` - *osp*
- `delete_directory_and_contents/1` - *osp*
- `delete_directory_contents/1` - *osp*
- `delete_edge/4` - *unweighted_graph_protocol*
- `delete_edge/5` - *weighted_graph_protocol*
- `delete_edges/3` - *graph_protocol*
- `delete_file/1` - *osp*
- `delete_in/4` - *nested_dictionary_protocol*
- `delete_matches/3` - *listp*
- `delete_max/4` - *dictionaryp*
- `delete_min/4` - *dictionaryp*
- `delete_object/4` - *s3_protocol*
- `delete_object/5` - *s3_session(HTTPTransport)*
- `delete_prefix/3` - *trie(Representation)*
- `delete_vertex/3` - *graph_protocol*
- `delete_vertices/3` - *graph_protocol*
- `dependents/1` - *packs*, *subject*
- `dependents/2` - *packs*
- `dependents/3` - *packs*
- `depth/2` - *termp*
- `depth_first_order/3` - *graph_protocol*
- `derangement/2` - *derangements_protocol*
- `derangement/3` - *derangements_protocol*
- `derangement_index/3` - *derangements_protocol*
- `derangement_index/4` - *derangements_protocol*

- derangements/2 - *derangements_protocol*
- derangements/3 - *derangements_protocol*
- derivative/2 - *univariate_function_protocol*
- derivative/3 - *ode_system_protocol*
- derivative/4 - *cubic_spline_interpolator*
- descendant/1 - *hierarchyp*
- descendant_class/1 - *class_hierarchyp*
- descendant_classes/1 - *class_hierarchyp*
- descendant_instance/1 - *class_hierarchyp*
- descendant_instances/1 - *class_hierarchyp*
- descendants/1 - *hierarchyp*
- describe/1 - *packs*, *registries*
- describe/2 - *packs*
- description/1 - *application_protocol*, *pack_protocol*, *registry_protocol*
- description/2 - *gravatar*
- destination_point/4 - *geospatial_protocol*
- destroy/0 - *amqp_pool*
- destroy/1 - *http_server_core_session*
- detach_sse_stream/0 - *mcp_server_streamable_http_transport*
- detect/2 - *language_detector_protocol*
- detect/3 - *language_detector_protocol*
- detect/4 - *language_detector_protocol*
- detect_all/2 - *language_detector_protocol*
- detect_all/3 - *language_detector_protocol*
- determinant/2 - *linear_algebra_protocol*
- deterministic/1 - *lgtunit*
- deterministic/2 - *lgtunit*
- diacritic_fold/2 - *text_normalization_profile_protocol*
- diagnostic/2 - *anomaly_detector_protocol*, *association_rule_miner_protocol*, *classifier_protocol*, *clusterer_protocol*, *dimension_reducer_protocol*, *feature_selector_protocol*, *forecaster_protocol*, *pattern_miner_protocol*, *ranker_protocol*, *recommender_protocol*, *regressor_protocol*, *text_vectorizer_protocol*, *tool_diagnostics_protocol*
- diagnostic/3 - *tool_diagnostics_protocol*
- diagnostic_rule/5 - *tool_diagnostics_protocol*
- diagnostic_rules/1 - *tool_diagnostics_protocol*
- diagnostic_target/1 - *tool_diagnostics_protocol*
- diagnostics/2 - *anomaly_detector_protocol*, *association_rule_miner_protocol*, *classifier_protocol*, *clusterer_protocol*, *dimension_reducer_protocol*, *feature_selector_protocol*, *forecaster_protocol*, *pattern_miner_protocol*, *ranker_protocol*, *recommender_protocol*, *regressor_protocol*, *text_vectorizer_protocol*, *tool_diagnostics_protocol*
- diagnostics/3 - *tool_diagnostics_protocol*
- diagnostics_preflight/2 - *tool_diagnostics_protocol*
- diagnostics_preflight/3 - *tool_diagnostics_protocol*
- diagnostics_summary/2 - *tool_diagnostics_protocol*
- diagnostics_summary/3 - *tool_diagnostics_protocol*
- diagnostics_tool/5 - *tool_diagnostics_protocol*
- diagonal_matrix/2 - *linear_algebra_protocol*
- diagonal_matrix/3 - *linear_algebra_protocol*
- diagonal_matrix/4 - *linear_algebra_protocol*
- diagram_description/1 - *diagram(Format)*
- diagram_name_suffix/1 - *diagram(Format)*
- dif/1 - *coroutining*, *dif*

- dif/2 - *coroutining*, *dif*
- difference_bounds/5 - *stn_protocol*
- difference_norm/3 - *linear_algebra_protocol*
- digest/2 - *hash_digest_protocol*
- digest/4 - *cmac_protocol*, *hmac_protocol*
- digest/5 - *cmac_protocol*, *hmac_protocol*
- digest_size/1 - *hash_digest_protocol*
- digit//1 - *number_grammars(Format)*
- digits//1 - *number_grammars(Format)*
- dimension_reducer_options/2 - *dimension_reducer_protocol*
- directories/1 - *lgtdocp*, *wrapper*
- directories/2 - *diagram(Format)*, *diagrams(Format)*, *lgtdocp*, *wrapper*
- directories/3 - *diagram(Format)*, *diagrams(Format)*
- directory/1 - *code_metric*, *dead_code_scanner*, *diagram(Format)*, *diagrams(Format)*, *lgtdocp*, *mutation_testing*, *packs_common*, *wrapper*
- directory/2 - *code_metric*, *dead_code_scanner*, *diagram(Format)*, *diagrams(Format)*, *lgtdocp*, *mutation_testing*, *packs_common*, *wrapper*
- directory/3 - *diagram(Format)*, *diagrams(Format)*
- directory_exists/1 - *osp*
- directory_files/2 - *osp*
- directory_files/3 - *osp*
- directory_mutants/2 - *mutation_testing*
- directory_mutants/3 - *mutation_testing*
- directory_score/2 - *code_metric*
- dirichlet/2 - *multivariate_distributions_protocol*, *sampling_protocol*
- dirichlet_density/3 - *multivariate_distributions_protocol*
- dirichlet_log_density/3 - *multivariate_distributions_protocol*
- dirichlet_samples/3 - *multivariate_distributions_protocol*
- disable/0 - *linter_reporter*
- disable/1 - *debug_messages*
- disable/2 - *debug_messages*
- disable_logging/1 - *loggingp*
- disconnect/1 - *memcached*, *redis*, *smtp_protocol*
- disconnect/2 - *mqtt*, *stomp*
- discover/3 - *open_id*, *open_id_client*
- discovery_url/2 - *open_id_discovery*
- disjoint/2 - *setp*
- disjoint_sets/2 - *union_find_protocol*
- dispatch/3 - *http_server_core*
- dispatch_map_id/2 - *ccsds_tc_services*
- dispatch_service_units/2 - *ccsds_tc_services*
- dispatch_service_units_by_map/2 - *ccsds_tc_services*
- dispatch_service_units_by_map/3 - *ccsds_tc_services*
- display_name/2 - *gravatar*
- distance/4 - *geospatial_protocol*, *stn_protocol*
- distance/5 - *geospatial_protocol*, *stn_protocol*
- distinct_arrangement/3 - *arrangements_protocol*
- distinct_arrangement/4 - *arrangements_protocol*
- distinct_arrangement_index/4 - *arrangements_protocol*

- distinct_arrangements/3 - *arrangements_protocol*
- distinct_arrangements/4 - *arrangements_protocol*
- distinct_cartesian_product/2 - *cartesian_products_protocol*
- distinct_cartesian_product/3 - *cartesian_products_protocol*
- distinct_cartesian_tuple/2 - *cartesian_products_protocol*
- distinct_cartesian_tuple/3 - *cartesian_products_protocol*
- distinct_cartesian_tuple_index/3 - *cartesian_products_protocol*
- distinct_combination/3 - *combinations_protocol*
- distinct_combination/4 - *combinations_protocol*
- distinct_combination_index/4 - *combinations_protocol*
- distinct_combinations/3 - *combinations_protocol*
- distinct_combinations/4 - *combinations_protocol*
- distinct_derangement/2 - *derangements_protocol*
- distinct_derangement/3 - *derangements_protocol*
- distinct_derangement_index/3 - *derangements_protocol*
- distinct_derangements/2 - *derangements_protocol*
- distinct_derangements/3 - *derangements_protocol*
- distinct_multiset/3 - *multisets_protocol*
- distinct_multiset/4 - *multisets_protocol*
- distinct_multiset_index/4 - *multisets_protocol*
- distinct_multisets/3 - *multisets_protocol*
- distinct_multisets/4 - *multisets_protocol*
- distinct_partition/2 - *integer_partitions_protocol*, *partitions_protocol*
- distinct_partition/3 - *integer_partitions_protocol*, *partitions_protocol*
- distinct_partition/4 - *integer_partitions_protocol*, *partitions_protocol*
- distinct_partition_index/3 - *integer_partitions_protocol*, *partitions_protocol*
- distinct_partition_index/4 - *integer_partitions_protocol*, *partitions_protocol*
- distinct_partition_index/5 - *integer_partitions_protocol*, *partitions_protocol*
- distinct_partitions/2 - *integer_partitions_protocol*, *partitions_protocol*
- distinct_partitions/3 - *integer_partitions_protocol*, *partitions_protocol*
- distinct_partitions/4 - *integer_partitions_protocol*, *partitions_protocol*
- distinct_permutation/2 - *permutations_protocol*
- distinct_permutation/3 - *permutations_protocol*
- distinct_permutation_index/3 - *permutations_protocol*
- distinct_permutations/2 - *permutations_protocol*
- distinct_permutations/3 - *permutations_protocol*
- distribution/1 - *application_protocol*
- diversity/2 - *genetic_algorithm_problem_protocol*
- doc_goal/1 - *doclet*
- document/1 - *open_ai*, *sbom*
- document/2 - *open_api*, *sbom*
- document/3 - *http_oauth_metadata*
- document/4 - *http_oauth_metadata*
- dot//1 - *number_grammars(Format)*
- dot_product/3 - *linear_algebra_protocol*
- double_metaphone/3 - *string_distance(Representation)*
- double_metaphone_match/2 - *string_distance(Representation)*
- dowhile/2 - *loopp*
- drop/3 - *listp*

- duration_between/3 - *datep*
- duration_string/2 - *iso8601*
- during/2 - *interval_protocol*
- dut1_entries/1 - *time_scales_data*, *time_scales_protocol*
- dut1_offset_at_utc_unix/3 - *time_scales_data*
- dut1_source/1 - *time_scales_data*, *time_scales_protocol*
- earliest/3 - *stn_protocol*
- earliest_schedule/2 - *stn_protocol*
- easter_day/3 - *iso8601*
- ed25519_keypair/2 - *ed25519*
- ed25519_public_key/2 - *ed25519*
- ed25519_sign/3 - *ed25519*
- ed25519_verify/3 - *ed25519*
- edge/3 - *unweighted_graph_protocol*
- edge/4 - *weighted_graph_protocol*
- edge/5 - *json_graph_data_protocol*
- edge/6 - *graph_language_protocol*
- edge_case/2 - *arbitrary*
- edges/2 - *graph_protocol*
- edit_similarity/3 - *string_distance(Representation)*
- edit_similarity/4 - *string_distance(Representation)*
- either/3 - *expected(Expected)*
- elicit_request/5 - *mcp_server_2025_06_18_spec*
- elicit_url_request/5 - *mcp_server_2025_11_25_spec*
- email_hash/2 - *gravatar*
- emit_progress/5 - *mcp_server_streamable_http_transport*
- emit_progress/6 - *mcp_server_2026_07_28_spec*
- empty/1 - *atms_environment_protocol*, *deque_protocol*, *dictionaryp*, *graph_protocol*, *heap_protocol*, *interval_relation_set_protocol*, *listp*, *lru_cache_protocol*, *nested_dictionary_protocol*, *optional*, *queuep*, *setp*, *trie(Representation)*, *varlistp*
- enable/0 - *linter_reporter*
- enable/1 - *debug_messages*, *linter_reporter*
- enable/2 - *debug_messages*
- enable_logging/1 - *loggingp*
- enabled/1 - *debug_messages*
- enabled/2 - *debug_messages*
- encode/2 - *json_rpc*, *sqids(Representation)*
- encode/3 - *geohash_protocol*, *sqids(Representation)*
- encode_bits/3 - *geohash_protocol*
- encode_body/4 - *http_body_codec_protocol*, *http_core*
- encode_frame/2 - *amqp*
- encode_int/3 - *geohash_protocol*
- encode_packet/2 - *mqtt*
- encoding_suffix/2 - *mime_types*
- encrypt/4 - *ecb*, *paseto*
- encrypt/5 - *cbc*
- encrypt_block/3 - *block_cipher_protocol*
- encrypt_padded/4 - *ecb*
- encrypt_padded/5 - *cbc*

- encrypt_prepared_block/3 - *block_cipher_prepared_key_protocol*
- ensure/2 - *http_server_core_session*
- ensure_directory/1 - *osp*
- ensure_file/1 - *osp*
- entailment_explanations/5 - *interval_constraint_network_protocol*
- entails/4 - *interval_constraint_network_protocol*, *stn_protocol*
- entails/5 - *interval_constraint_network_protocol*, *stn_protocol*
- entity/1 - *code_metric*, *dead_code_scanner*, *help*, *mutation_testing*, *xref_diagram(Format)*
- entity/2 - *dead_code_scanner*, *mutation_testing*, *xref_diagram(Format)*
- entity_info_pair_score_hook/3 - *doc_metric*
- entity_info_score_hook/2 - *doc_metric*
- entity_mutants/2 - *mutation_testing*
- entity_mutants/3 - *mutation_testing*
- entity_predicates_weights_hook/2 - *doc_metric*
- entity_prefix/2 - *logtalk*
- entity_score/2 - *code_metric*
- entry/1 - *crontab(Representation)*
- enumerate/2 - *random_protocol*
- environment_variable/2 - *osp*
- epoch/1 - *ccsds_time_code_protocol*
- epoch/2 - *ccsds_time_fields*
- epsg_crs/2 - *crs_projections_protocol*
- epsilon/1 - *lgtunit*
- equal/2 - *atms_environment_protocol*, *interval_protocol*, *setp*
- equality_constraints/2 - *constrained_optimization_problem_protocol*
- equality_jacobian/2 - *constrained_optimization_problem_protocol*
- equirectangular_inverse/4 - *geospatial_protocol*
- equirectangular_projection/4 - *geospatial_protocol*
- equivalent/2 - *interval_constraint_network_protocol*, *url(Representation)*
- erase/1 - *recorded_database_core*
- error/2 - *json_rpc*
- error_code/2 - *json_rpc*
- error_data/2 - *json_rpc*
- error_message/2 - *json_rpc*
- error_response/2 - *s3_xml*
- error_response/4 - *json_rpc*
- error_response/5 - *json_rpc*
- es256_signature_der/2 - *jwt_der*, *open_id_der*
- essentially_equal/3 - *lgtunit*, *number*
- estimate_temperature/1 - *simulated_annealing(Problem,Random)*
- estimate_temperature/2 - *simulated_annealing(Problem,Random)*
- euclidean_distance/3 - *numberlistp*
- euclidean_norm/2 - *linear_algebra_protocol*, *numberlistp*
- evaluate/2 - *univariate_function_protocol*
- evaluate/3 - *interpolator_protocol*, *json_path_protocol*, *json_pointer_protocol*
- evaluate_relative/4 - *json_pointer_protocol*
- event/2 - *open_ai_event_stream*
- event_interval/4 - *stn_protocol*
- event_relation/6 - *stn_protocol*

- `evict/3` - *lru_cache_protocol*
- `exactly_representable/1` - *ieee_754_protocol*
- `example/2` - *clustering_dataset_protocol*, *dimension_reduction_dataset_protocol*
- `example/3` - *anomaly_dataset_protocol*, *dataset_protocol*, *feature_dataset_protocol*, *regression_dataset_protocol*, *supervised_dimension_reduction_dataset_protocol*, *target_supervised_dimension_reduction_dataset_protocol*
- `example_count/1` - *feature_dataset_protocol*
- `exchange/3` - *http_transport_protocol*
- `exchange/4` - *http_client_core*, *http_transport_protocol*
- `exchange/5` - *http_transport_protocol*
- `exchange_bind/4` - *amqp*
- `exchange_code/5` - *open_id*, *open_id_client*
- `exchange_declare/3` - *amqp*
- `exchange_delete/3` - *amqp*
- `exchange_sequence/3` - *http_transport_protocol*
- `exchange_sequence/4` - *http_client_core*, *http_transport_protocol*
- `exchange_unbind/4` - *amqp*
- `exclude/2` - *stop_words_protocol*
- `exclude/3` - *metap*
- `excluded/4` - *interval_constraint_network_protocol*
- `execution_context/7` - *logtalk*
- `exists/3` - *redis*
- `expand/2` - *geohash_protocol*, *json_ld_protocol*
- `expand/3` - *uri_template(Representation)*
- `expand/4` - *uri_template(Representation)*
- `expand_int/3` - *geohash_protocol*
- `expand_library_path/2` - *logtalk*
- `expected/1` - *expected(Expected)*
- `expecteds/2` - *either*
- `expire/4` - *redis*
- `explain/2` - *datalog_protocol*
- `explain//1` - *tutor_explanations*
- `exponential/2` - *sampling_protocol*, *univariate_distributions_protocol*
- `exponential_density/3` - *univariate_distributions_protocol*
- `exponential_distribution/3` - *univariate_distributions_protocol*
- `exponential_log_density/3` - *univariate_distributions_protocol*
- `exponential_quantile/3` - *univariate_distributions_protocol*
- `exponential_samples/3` - *univariate_distributions_protocol*
- `export/1` - *sbom*
- `export/2` - *sbom*
- `export_to_clauses/4` - *anomaly_detector_protocol*, *association_rule_miner_protocol*, *classifier_protocol*, *clusterer_protocol*, *dimension_reducer_protocol*, *feature_selector_protocol*, *forecaster_protocol*, *pattern_miner_protocol*, *ranker_protocol*, *recommender_protocol*, *regressor_protocol*
- `export_to_file/4` - *anomaly_detector_protocol*, *association_rule_miner_protocol*, *classifier_protocol*, *clusterer_protocol*, *dimension_reducer_protocol*, *feature_selector_protocol*, *forecaster_protocol*, *pattern_miner_protocol*, *ranker_protocol*, *recommender_protocol*, *regressor_protocol*
- `extend_catalog/3` - *bm25_recommender*, *jaccard_recommender*, *tfidf_recommender*
- `extension/1` - *proto_hierarchyp*
- `extension_type/2` - *mime_types*
- `extension_type/3` - *mime_types*
- `extensions/1` - *proto_hierarchyp*

- external_reference/2 - *application_protocol*
- extract_aos_packets/3 - *ccsds_packet_services*
- extract_packets/3 - *ccsds_frames*
- extract_packets/4 - *ccsds_link_profiles*
- extract_tc_segment/2 - *ccsds_tc_services*
- extract_tm_packets/3 - *ccsds_packet_services*
- factorial/2 - *natural*
- facts/1 - *datalog_protocol*
- failed_test_reason//1 - *lgtunit_messages*
- failure_details/3 - *smtp_process_transport*
- falling_factorial/3 - *natural*
- false/1 - *java_utils_protocol*
- fcube/0 - *fcube*
- feature_scores/2 - *feature_selector_protocol*
- fecf/2 - *ccsds_frame_protocol*, *ccsds_frames*
- fibonacci/2 - *natural*
- field/3 - *gravatar*
- field/4 - *http_multipart*
- field_part/4 - *http_multipart*
- fields/2 - *http_multipart*, *nmea_protocol*
- fields/5 - *ieee_754_fields_protocol*
- file/1 - *code_metric*, *dead_code_scanner*, *entity_diagram(Format)*, *lgtdocp*, *wrapper*
- file/2 - *code_metric*, *dead_code_scanner*, *entity_diagram(Format)*, *lgtdocp*, *wrapper*
- file/6 - *http_multipart*
- file_exists/1 - *osp*
- file_footer/3 - *graph_language_protocol*
- file_header/3 - *graph_language_protocol*
- file_modification_time/2 - *osp*
- file_part/6 - *http_multipart*
- file_path_components/2 - *url(Representation)*
- file_permission/2 - *osp*
- file_score/2 - *code_metric*
- file_size/2 - *osp*
- file_to_bytes/2 - *reader*
- file_to_bytes/3 - *reader*
- file_to_chars/2 - *reader*
- file_to_chars/3 - *reader*
- file_to_codes/2 - *reader*
- file_to_codes/3 - *reader*
- file_to_terms/2 - *reader*
- file_to_terms/3 - *reader*
- file_type_extension/2 - *logtalk*
- files/1 - *diagram(Format)*, *diagrams(Format)*, *lgtdocp*, *wrapper*
- files/2 - *diagram(Format)*, *diagrams(Format)*, *http_multipart*, *lgtdocp*, *wrapper*
- files/3 - *diagram(Format)*, *diagrams(Format)*
- filter/2 - *optional(Optional)*
- filter/3 - *expected(Expected)*, *validation(Validation)*
- final/2 - *http_websocket_frames*
- final_bearing/3 - *geospatial_protocol*

- `final_hash_state/2` - *hash_state_protocol*
- `find/4` - *union_find_protocol*
- `find/5` - *union_find_protocol*
- `find_root/2` - *root_finder_protocol*
- `find_root/3` - *root_finder_protocol*
- `find_root/4` - *root_finder_protocol*
- `findall_in_noblock/3` - *linda_client*
- `findall_in_noblock/4` - *linda_client*
- `findall_member/4` - *metap*
- `findall_member/5` - *metap*
- `findall_rd_noblock/3` - *linda_client*
- `findall_rd_noblock/4` - *linda_client*
- `fine/2` - *ccsds_cuc(CoarseOctets,FineOctets,Epoch)*
- `finish/3` - *http_server_core_session*
- `finished_by/2` - *interval_protocol*
- `finishes/2` - *interval_protocol*
- `finite_binary_rational/4` - *ieee_754_fields_protocol*
- `first_header_pointer/2` - *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- `first_significant_component/2` - *linear_algebra_protocol*
- `first_significant_component/3` - *linear_algebra_protocol*
- `fisher/3` - *sampling_protocol*, *univariate_distributions_protocol*
- `fisher_density/4` - *univariate_distributions_protocol*
- `fisher_distribution/4` - *univariate_distributions_protocol*
- `fisher_log_density/4` - *univariate_distributions_protocol*
- `fisher_quantile/4` - *univariate_distributions_protocol*
- `fisher_samples/4` - *univariate_distributions_protocol*
- `fit/2` - *interpolator_protocol*
- `fit/3` - *interpolator_protocol*
- `fitness/2` - *differential_evolution_problem_protocol*, *particle_swarm_optimization_protocol*
- `fitted_values/2` - *intermittent_demand_forecasting*, *theta_forecasting*
- `fitted_values/3` - *intermittent_demand_forecasting*, *theta_forecasting*
- `flag_group_chk/1` - *flags*
- `flag_groups/1` - *flags*
- `flat_map/2` - *expected(Expected)*, *optional(Optional)*, *validation(Validation)*
- `flatten/1` - *expected(Expected)*, *optional(Optional)*, *validation(Validation)*
- `flatten/2` - *json_ld_protocol*, *listp*, *varlistp*
- `float//1` - *number_grammars(Format)*
- `flow_balances/3` - *flow_network_protocol*
- `flow_cost/3` - *flow_network_protocol*
- `flush_all/1` - *memcached*
- `flush_all/2` - *memcached*
- `fold_diacritics/2` - *text_normalizer_protocol*
- `fold_left/4` - *metap*
- `fold_left_1/3` - *metap*
- `fold_right/4` - *metap*
- `fold_right_1/3` - *metap*
- `footer/2` - *paseto_protocol*
- `forbidden_response/3` - *http_oauth*
- `forbidden_response/4` - *http_oauth*

- fordownto/3 - *loopp*
- fordownto/4 - *loopp*
- fordownto/5 - *loopp*
- foreach/3 - *loopp*
- foreach/4 - *loopp*
- forecast/3 - *forecaster_protocol*
- forecast_interval/5 - *exponential_smoothing*, *time_series_regression*
- forecaster_options/2 - *forecaster_protocol*
- form_body/2 - *rest*
- form_data_body/2 - *http_multipart*
- format/1 - *ccsds_time_code_protocol*
- format/2 - *ccsds_time_codes*, *ccsds_time_fields*, *format*
- format/3 - *format*
- format_date_time/4 - *datep*
- format_entity_score//2 - *code_metric*
- format_object/1 - *diagram(Format)*
- format_report/1 - *mutation_testing*
- format_report/2 - *mutation_testing*
- format_report/3 - *mutation_testing*
- format_to_atom/3 - *term_io_protocol*
- format_to_chars/3 - *term_io_protocol*
- format_to_chars/4 - *term_io_protocol*
- format_to_codes/3 - *term_io_protocol*
- format_to_codes/4 - *term_io_protocol*
- formatted_iban/2 - *iso_13616_protocol*
- forto/3 - *loopp*
- forto/4 - *loopp*
- forto/5 - *loopp*
- forward/1 - *forwarding*
- forward/2 - *zipperp*
- forward/3 - *zipperp*
- forward_substitution/3 - *linear_algebra_protocol*
- fractile/3 - *statisticsp*
- frame/2 - *open_ai_event_stream*
- frame/3 - *json_ld_protocol*
- frame/5 - *http_websocket_frames*
- frame_body/2 - *stomp*
- frame_command/2 - *stomp*
- frame_header/3 - *stomp*
- frame_headers/2 - *stomp*
- frame_type/2 - *ccsds_frame_protocol*, *ccsds_frames*
- freeze/2 - *coroutining*
- frequency/1 - *time_series_dataset_protocol*
- frequency_distribution/2 - *statisticsp*
- frobenius_norm/2 - *linear_algebra_protocol*
- from_expected/2 - *validation*
- from_generator/2 - *expected*, *optional*, *validation*
- from_generator/3 - *expected*, *optional*, *validation*
- from_generator/4 - *expected*, *validation*

- `from_goal/2` - *expected*, *optional*, *validation*
- `from_goal/3` - *expected*, *optional*, *validation*
- `from_goal/4` - *expected*, *validation*
- `from_goal_or_throw/2` - *optional*
- `from_goal_or_throw/3` - *optional*
- `from_graph/2` - *flow_network_protocol*
- `from_list/2` - *atms_environment_protocol*
- `from_optional/3` - *expected*, *validation*
- `from_unix_seconds/2` - *ccsds_time_code_protocol*
- `from_uuid/3` - *typeid_protocol*
- `frozen/2` - *coroutining*
- `full_device_path/1` - *osp*
- `func_test/3` - *metagol*
- `functional/0` - *metagol*
- `fund_currency/5` - *iso_4217*, *iso_4217_protocol*
- `game/4` - *temporal_pairwise_ranking_dataset_protocol*
- `gamma/3` - *sampling_protocol*, *univariate_distributions_protocol*
- `gamma_density/4` - *univariate_distributions_protocol*
- `gamma_distribution/4` - *univariate_distributions_protocol*
- `gamma_log_density/4` - *univariate_distributions_protocol*
- `gamma_quantile/4` - *univariate_distributions_protocol*
- `gamma_samples/4` - *univariate_distributions_protocol*
- `gat/4` - *memcached*
- `gats/5` - *memcached*
- `gc/2` - *http_server_core_session*
- `generate/1` - *cuid2_protocol*, *ids(Representation,Bytes)*, *ksuid_protocol*, *nanoid_protocol*, *snowflakeid_protocol*, *typeid_protocol*, *ulid_protocol*
- `generate/2` - *base32*, *base58*, *base64*, *base64url*, *base64url_no_padding*, *base85*, *bson(StringRepresentation)*, *cbor(StringRepresentation)*, *ccsds_frame_protocol*, *ccsds_packets(SecondaryHeaderLength)*, *ccsds_time_code_protocol*, *crontab(Representation)*, *geojson_protocol*, *gpx_protocol*, *html*, *http_websocket_frames*, *ieee_754_protocol*, *json_graph_protocol*, *json_ld_protocol*, *json_lines_protocol*, *json_path_protocol*, *json_pointer_protocol*, *json_protocol*, *message_pack(StringRepresentation)*, *open_api*, *sarif*, *toml_protocol*, *toon_protocol*, *typeid_protocol*, *ulid_protocol*, *url(Representation)*, *wkt_wkb_protocol*, *yaml_protocol*
- `generate/3` - *avro*, *ccsds_frame_protocol*, *ccsds_packets(SecondaryHeaderLength)*, *ccsds_time_code_protocol*, *ccsds_time_fields*, *crontab(Representation)*, *http_multipart*, *ieee_754_protocol*, *protobuf*, *typeid_protocol*
- `generate/4` - *avro*, *protobuf*, *sarif*
- `generate/8` - *ulid_protocol*
- `generate_all/2` - *yaml_protocol*
- `generate_authentication_info/2` - *http_digest*
- `generate_authorization/2` - *http_authenticate*, *http_digest*, *http_oauth*
- `generate_body/3` - *http_core*
- `generate_challenge/2` - *http_authenticate*, *http_digest*, *http_oauth*
- `generate_cookie/2` - *http_cookies(Representation)*
- `generate_fragment/2` - *json_pointer_protocol*
- `generate_frame/3` - *ccsds_link_profiles*
- `generate_frames/3` - *ccsds_link_profiles*
- `generate_headers/2` - *http_core*
- `generate_idle_packet/4` - *ccsds_packetization*
- `generate_relative/2` - *json_pointer_protocol*
- `generate_request/2` - *http_core*

- generate_request_line/2 - *http_core*
- generate_response/2 - *http_core*
- generate_response_headers/2 - *http_core*
- generate_set_cookie/4 - *http_cookies(Representation)*
- generate_status_line/2 - *http_core*
- genint/2 - *genint_core*
- gensym/2 - *gensym_core*
- geohash_to_int/2 - *geohash_protocol*
- geojson_to_json/2 - *geojson_protocol*
- geometric/2 - *sampling_protocol*
- geometric_mean/2 - *statisticsp*
- get/1 - *optional(Optional)*
- get/3 - *generalized_ripple_down_rules*, *http_client*, *http_server_core_session*, *memcached*, *redis*
- get/4 - *http_client*, *http_client_digest_session(HTTPTransport)*, *http_client_session(HTTPTransport)*, *lru_cache_protocol*, *memcached*
- get_field/2 - *java_access_protocol*
- get_flag_value/2 - *flags*
- get_object/5 - *s3_protocol*
- get_object/6 - *s3_session(HTTPTransport)*
- get_seed/1 - *arbitrary*, *pseudo_random_protocol*
- getrange/5 - *redis*
- gets/4 - *memcached*
- gets/5 - *memcached*
- git_object_identifier/1 - *application_protocol*
- gnu/0 - *fcube*
- goal_expansion/2 - *expanding*
- gpx_to_xml/2 - *gpx_protocol*
- gradient/2 - *local_optimization_problem_protocol*
- gram_matrix/2 - *linear_algebra_protocol*
- graph/2 - *json_graph_data_protocol*
- graph_coloring/3 - *undirected_graph_common*
- graph_footer/5 - *graph_language_protocol*
- graph_header/5 - *graph_language_protocol*
- ground/1 - *termp*
- ground_track/5 - *tle_orbits_protocol*
- ground_track/6 - *tle_orbits_protocol*
- group/1 - *ranking_dataset_protocol*
- group_by_key/2 - *pairs*
- group_consecutive_by_key/2 - *pairs*
- group_sorted_by_key/2 - *pairs*
- grouped_dataset_groups/2 - *ranking_dataset_common*
- grouped_dataset_items/2 - *ranking_dataset_common*
- grouped_dataset_items/3 - *ranking_dataset_common*
- grouped_dataset_summary/2 - *ranking_dataset_common*
- guess_all_extensions/2 - *mime_types*
- guess_all_extensions/3 - *mime_types*
- guess_arity/2 - *csv_protocol*
- guess_extension/2 - *mime_types*
- guess_extension/3 - *mime_types*

- guess_file_type/3 - *mime_types*
- guess_file_type/4 - *mime_types*
- guess_separator/2 - *csv_protocol*
- guess_type/3 - *mime_types*
- guess_type/4 - *mime_types*
- gumbel/3 - *sampling_protocol*
- ha1/4 - *http_digest_verifier_protocol*
- halt/0 - *logtalk*
- halt/1 - *logtalk*
- halt_hook/2 - *logtalk*
- hamming/3 - *string_distance(Representation)*
- hamming_distance/3 - *listp*
- handbook/0 - *help*
- handle/2 - *http_handler_protocol*, *http_router*, *http_sse_service_handler_protocol*, *http_websocket_service_handler_protocol*
- handle_mcp_request/4 - *mcp_server_streamable_http_transport*
- handle_message/3 - *mcp_server_spec_protocol*
- handle_open_ai/3 - *open_ai_backend_protocol*
- harmonic_mean/2 - *statisticsp*
- has_cycle/1 - *directed_graph_protocol*, *undirected_graph_common*
- has_negative_cycle/1 - *weighted_graph_protocol*
- has_path/3 - *graph_protocol*
- hash/2 - *gravatar*, *hash_protocol*
- haversine_distance/3 - *geospatial_protocol*
- hdel/4 - *redis*
- head/2 - *queuep*
- head/3 - *http_client*
- head/4 - *http_client*, *http_client_digest_session(HTTPTransport)*, *http_client_session(HTTPTransport)*
- head_bucket/3 - *s3_protocol*
- head_bucket/4 - *s3_session(HTTPTransport)*
- head_object/4 - *s3_protocol*
- head_object/5 - *s3_session(HTTPTransport)*
- head_pred/1 - *metagol*
- header/2 - *http_request_protocol*, *http_response_protocol*, *jwt*, *jwt_compact*
- header/3 - *http_core*
- header_algorithm/2 - *jwt_jwa*, *open_id_jwks*
- header_key_id/2 - *jwt_jwa*, *open_id_jwks*
- header_value/3 - *s3_headers*
- headers/1 - *http_request_protocol*, *http_response_protocol*
- headers/2 - *http_core*
- help/0 - *help*, *packs_common*
- help/1 - *command_line_option*
- help/2 - *command_line_options*
- help/3 - *command_line_options*
- hessian/2 - *local_optimization_problem_protocol*
- heuristic/3 - *ant_colony_problem_protocol*
- hex_bytes/2 - *crypto*
- hex_digest/4 - *cmac_protocol*, *hmac_protocol*
- hex_digest/5 - *cmac_protocol*, *hmac_protocol*
- hex_digit//1 - *number_grammars(Format)*

- hex_digits//1 - *number_grammars(Format)*
- hexists/4 - *redis*
- hget/4 - *redis*
- hgetall/3 - *redis*
- hkdf/5 - *crypto*
- hkeys/3 - *redis*
- hlen/3 - *redis*
- hmac_hash/2 - *jwt_jwa*
- home/1 - *pack_protocol*, *registry_protocol*
- homepage/1 - *application_protocol*
- hotp/5 - *otp_protocol*
- hotp_verify/7 - *otp_protocol*
- hset/5 - *redis*
- hvals/3 - *redis*
- hyperedge/4 - *json_graph_data_protocol*
- hyperedge/5 - *json_graph_data_protocol*
- hypergeometric/4 - *sampling_protocol*
- iban/4 - *iso_13616_protocol*
- ibk/3 - *metagol*
- id/2 - *json_rpc*
- identity_matrix/2 - *linear_algebra_protocol*
- if_empty/1 - *optional(Optional)*
- if_expected/1 - *expected(Expected)*
- if_expected_or_else/2 - *expected(Expected)*
- if_invalid/1 - *validation(Validation)*
- if_present/1 - *optional(Optional)*
- if_present_or_else/2 - *optional(Optional)*
- if_unexpected/1 - *expected(Expected)*
- if_valid/1 - *validation(Validation)*
- if_valid_or_else/2 - *validation(Validation)*
- in/1 - *linda_client*
- in/2 - *linda_client*
- in_degree/3 - *directed_graph_protocol*
- in_label/3 - *atms_protocol*
- in_list/2 - *linda_client*
- in_list/3 - *linda_client*
- in_noblock/1 - *linda_client*
- in_noblock/2 - *linda_client*
- include/3 - *metap*
- incr/3 - *redis*
- incr/4 - *memcached*
- incrby/4 - *redis*
- increase/1 - *counter*
- increment/0 - *counter*
- increment_counter/1 - *counters*
- inequality_constraints/2 - *constrained_optimization_problem_protocol*
- inequality_jacobian/2 - *constrained_optimization_problem_protocol*
- init/0 - *shell(Interpreters)*
- init/2 - *subsequences_protocol*

- `init1/2` - *subsequences_protocol*
- `init_log_file/2` - *loggingp*
- `init_tail/2` - *subsequences_protocol*
- `init_tails/2` - *subsequences_protocol*
- `initial_bearing/3` - *geospatial_protocol*
- `initial_channel_reassembly_state/1` - *ccsds_packet_services*
- `initial_point/1` - *local_optimization_problem_protocol*
- `initial_population/1` - *genetic_algorithm_problem_protocol*
- `initial_positions/1` - *differential_evolution_problem_protocol*, *particle_swarm_optimization_protocol*
- `initial_reassembly_state/1` - *ccsds_link_profiles*, *ccsds_packet_services*, *ccsds_tc_services*
- `initial_state/1` - *ccsds_packetization*, *http_websocket_session(Role,TextRepresentation)*, *simulated_annealing_protocol*, *tabu_search_problem_protocol*
- `initial_temperature/1` - *simulated_annealing_protocol*
- `initial_velocities/1` - *particle_swarm_optimization_protocol*
- `initialize/1` - *amqp_pool*
- `inits/2` - *subsequences_protocol*
- `inits1/2` - *subsequences_protocol*
- `inner_progress/6` - *constrained_optimization_problem_protocol*
- `inorder/2` - *bintree*
- `input_schema/2` - *mcp_tool_protocol*
- `insert/3` - *setp*
- `insert/4` - *dictionaryp*, *heap_protocol*, *trie(Representation)*
- `insert_after/3` - *zipperp*
- `insert_all/3` - *heap_protocol*, *setp*
- `insert_aos_packets/4` - *ccsds_packet_services*
- `insert_before/3` - *zipperp*
- `insert_in/4` - *nested_dictionary_protocol*
- `insert_packets/4` - *ccsds_frames*
- `insert_packets/5` - *ccsds_link_profiles*
- `insert_tc_segment/3` - *ccsds_tc_services*
- `insert_tm_packets/4` - *ccsds_packet_services*
- `insert_zone/2` - *ccsds_aos_frames(FrameLength,InsertZoneLength,HasOCF,HasFECF)*
- `install/1` - *packs*
- `install/2` - *packs*
- `install/3` - *packs*
- `install/4` - *packs*
- `installed/0` - *packs*
- `installed/1` - *packs*
- `installed/3` - *packs*
- `installed/4` - *packs*
- `instance/1` - *class_hierarchyp*
- `instance/2` - *recorded_database_core*
- `instances/1` - *class_hierarchyp*
- `instant_to_utc_date_time/2` - *time_scales_protocol*
- `int_to_geohash/3` - *geohash_protocol*
- `integer//1` - *number_grammars(Format)*
- `integer_to_bytes/4` - *byte_order*
- `integer_to_bytes/5` - *byte_order*
- `integrate/3` - *quadrature_protocol*

- integrate/4 - *quadrature_protocol*
- integrate/5 - *quadrature_protocol*
- internal_error/2 - *json_rpc*
- internal_os_path/2 - *osp*
- interpolate_great_circle/4 - *geospatial_protocol*
- interpolate_rhumb/4 - *geospatial_protocol*
- interpretations/2 - *atms_protocol*
- interquartile_range/2 - *statisticsp*
- intersect/2 - *setp*
- intersection/2 - *dictionaryp*
- intersection/3 - *dictionaryp*, *interval_relation_set_protocol*, *setp*
- intersection/4 - *setp*
- interval_string/2 - *iso8601*
- introspect/4 - *http_oauth_introspection_client*
- invalid/1 - *validation(Validation)*
- invalid_params/2 - *json_rpc*
- invalid_request/1 - *json_rpc*
- invalids/2 - *validated*
- inverse_matrix/2 - *linear_algebra_protocol*
- invert_from_cholesky/2 - *linear_algebra_protocol*
- invoke/1 - *java_access_protocol*
- invoke/2 - *java_access_protocol*
- ipv4//1 - *ip_grammars(Format)*
- ipv6//1 - *ip_grammars(Format)*
- is_absolute_file_name/1 - *osp*
- is_acyclic/1 - *directed_graph_protocol*
- is_alpha/1 - *characterp*
- is_alphanumeric/1 - *characterp*
- is_ascii/1 - *characterp*
- is_batch/1 - *json_rpc*
- is_bin_digit/1 - *characterp*
- is_bipartite/1 - *graph_protocol*
- is_body/1 - *http_multipart*
- is_boosted_request/1 - *http_htmx*
- is_clockwise_polygon/1 - *geospatial_protocol*
- is_complete/1 - *graph_protocol*
- is_connected/1 - *undirected_graph_common*
- is_control/1 - *characterp*
- is_cors_request/1 - *http_cors*
- is_dec_digit/1 - *characterp*
- is_empty/0 - *optional(Optional)*
- is_end_of_line/1 - *characterp*
- is_error_response/1 - *json_rpc*
- is_expected/0 - *expected(Expected)*
- is_false/1 - *java_utils_protocol*
- is_fragment_request/1 - *http_htmx*
- is_frame/1 - *http_websocket_frames*
- is_hex_digit/1 - *characterp*
- is_history_restore_request/1 - *http_htmx*

- is_htmx_request/1 - *http_htmx*
- is_invalid/0 - *validation(Validation)*
- is_layout/1 - *characterp*
- is_letter/1 - *characterp*
- is_lower_case/1 - *characterp*
- is_message/1 - *http_websocket_messages(TextRepresentation)*
- is_newline/1 - *characterp*
- is_notification/1 - *json_rpc*
- is_null/1 - *java_utils_protocol*
- is_object/1 - *java_utils_protocol*
- is_octal_digit/1 - *characterp*
- is_part/1 - *http_multipart*
- is_period/1 - *characterp*
- is_prefix_of/2 - *subsequences_protocol*
- is_preflight_request/1 - *http_cors*
- is_present/0 - *optional(Optional)*
- is_punctuation/1 - *characterp*
- is_quote/1 - *characterp*
- is_request/1 - *http_core*, *json_rpc*
- is_response/1 - *http_core*, *json_rpc*
- is_sparse/1 - *graph_protocol*
- is_state/1 - *http_websocket_session(Role,TextRepresentation)*
- is_stop_word/1 - *stop_words_protocol*
- is_subsequence_of/2 - *subsequences_protocol*
- is_suffix_of/2 - *subsequences_protocol*
- is_tree/1 - *undirected_graph_common*
- is_true/1 - *java_utils_protocol*
- is_unexpected/0 - *expected(Expected)*
- is_upper_case/1 - *characterp*
- is_valid/0 - *validation(Validation)*
- is_valid_polygon/1 - *geospatial_protocol*
- is_void/1 - *java_utils_protocol*
- is_vowel/1 - *characterp*
- is_white_space/1 - *characterp*
- item/1 - *item_content_dataset_protocol*, *multiplayer_ranking_dataset_protocol*, *pairwise_measurement_dataset_protocol*, *pairwise_ranking_dataset_protocol*, *temporal_pairwise_ranking_dataset_protocol*
- item/2 - *ranking_dataset_protocol*
- item_content/2 - *item_content_dataset_protocol*
- items/1 - *sequence_dataset_protocol*, *transaction_dataset_protocol*
- iterator_element/2 - *java_utils_protocol*
- jaccard_index/3 - *string_distance(Representation)*
- jaro/3 - *string_distance(Representation)*
- jaro_winkler/3 - *string_distance(Representation)*
- job_title/2 - *gravatar*
- join/3 - *queuep*
- join_all/3 - *queuep*
- join_aos_packet_zone/3 - *ccsds_packet_services*
- join_packet_zone/4 - *ccsds_packet_services*
- json_array_body/2 - *rest*

- json_body/2 - *rest*
- json_media_descriptor/2 - *open_api*
- json_media_descriptor/3 - *open_api*
- json_object_body/2 - *rest*
- json_request_body_descriptor/4 - *open_api*
- json_request_body_descriptor/5 - *open_api*
- json_response/4 - *rest*
- json_response/5 - *rest*
- json_response_descriptor/4 - *open_api*
- json_response_descriptor/5 - *open_api*
- json_to_geojson/2 - *geojson_protocol*
- jump/3 - *queuep*
- jump_all/3 - *queuep*
- jump_all_block/3 - *queuep*
- justifications/2 - *atms_protocol*
- justify/5 - *atms_protocol*
- jwks/3 - *open_id*, *open_id_client*
- k_distinct_subsequence/3 - *subsequences_protocol*
- k_distinct_subsequences/3 - *subsequences_protocol*
- k_permutation/3 - *permutations_protocol*
- k_permutation/4 - *permutations_protocol*
- k_permutations/3 - *permutations_protocol*
- k_permutations/4 - *permutations_protocol*
- key/2 - *pairs*
- key_size/1 - *block_cipher_protocol*
- key_type/2 - *jwt_jwa*
- keys/2 - *dictionaryp*, *pairs*
- keys/3 - *redis*
- keys_values/3 - *pairs*
- keysort/2 - *listp*
- kill/1 - *process*
- kill/2 - *process*
- known_zone_id/1 - *tzif_zone_ids*
- kurtosis/2 - *statisticsp*
- label/3 - *atms_protocol*
- lambert_azimuthal_equal_area/3 - *crs_projections_protocol*
- lambert_azimuthal_equal_area_inverse/3 - *crs_projections_protocol*
- language/1 - *language_profile_protocol*
- language/3 - *iso_639_1*, *iso_639_protocol*
- language/5 - *iso_639_3*, *iso_639_protocol*
- language_code/5 - *iso_639_2*, *iso_639_protocol*
- language_group/2 - *iso_639_5*, *iso_639_protocol*
- language_object/2 - *graph_language_registry*
- language_scripts/2 - *language_detection_scripts*
- last/2 - *listp*, *varlistp*
- latest/3 - *stn_protocol*
- leaf/1 - *hierarchyp*
- leaf_class/1 - *class_hierarchyp*
- leaf_classes/1 - *class_hierarchyp*

- leaf_instance/1 - *class_hierarchyp*
- leaf_instances/1 - *class_hierarchyp*
- leap_effective_date/2 - *time_scales_data*
- leap_offset_at_utc_unix/2 - *time_scales_data*
- leap_second_date/2 - *time_scales_protocol*
- leap_seconds_entries/1 - *time_scales_data*, *time_scales_protocol*
- leap_seconds_source/1 - *time_scales_data*, *time_scales_protocol*
- leap_year/1 - *datep*, *iso8601*
- learn/0 - *metagol_example_protocol*
- learn/1 - *metagol_example_protocol*
- learn/2 - *anomaly_detector_protocol*, *classifier_protocol*, *clusterer_protocol*, *dimension_reducer_protocol*, *feature_selector_protocol*, *forecaster_protocol*, *metagol*, *ranker_protocol*, *recommender_protocol*, *regressor_protocol*, *text_vectorizer_protocol*
- learn/3 - *anomaly_detector_protocol*, *classifier_protocol*, *clusterer_protocol*, *dimension_reducer_protocol*, *feature_selector_protocol*, *forecaster_protocol*, *metagol*, *ranker_protocol*, *recommender_protocol*, *regressor_protocol*, *text_vectorizer_protocol*, *truncated_svd_projection*
- learn_seq/2 - *metagol*
- learn_transform/3 - *text_vectorizer_protocol*
- learn_transform/4 - *text_vectorizer_protocol*
- learn_with_timeout/4 - *metagol*
- leash/1 - *debuggerp*
- leashing/1 - *debuggerp*
- least_common_multiple/2 - *numberlistp*
- least_squares/3 - *linear_algebra_protocol*
- leaves/1 - *hierarchyp*
- leaves/2 - *unweighted_directed_graph(Dictionary)*
- lemma/2 - *lemmatizer_language_protocol*, *lemmatizer_protocol*
- lemma/3 - *lemmatizer_language_protocol*, *lemmatizer_protocol*
- lemmas/2 - *lemmatizer_protocol*
- lemmas/3 - *lemmatizer_protocol*
- length/2 - *deque_protocol*, *listp*, *queuep*, *varlistp*
- levenshtein/3 - *string_distance(Representation)*
- libraries/0 - *help*
- libraries/1 - *diagram(Format)*, *diagrams(Format)*, *lgtdocp*
- libraries/2 - *diagram(Format)*, *diagrams(Format)*, *lgtdocp*
- libraries/3 - *diagram(Format)*, *diagrams(Format)*
- library/1 - *code_metric*, *dead_code_scanner*, *diagram(Format)*, *diagrams(Format)*, *help*, *lgtdocp*, *mutation_testing*
- library/2 - *code_metric*, *dead_code_scanner*, *diagram(Format)*, *diagrams(Format)*, *lgtdocp*, *mutation_testing*
- library_mutants/2 - *mutation_testing*
- library_mutants/3 - *mutation_testing*
- library_score/2 - *code_metric*
- license/1 - *application_protocol*, *pack_protocol*
- linda/0 - *linda_server*
- linda/1 - *linda_server*
- linda_client/1 - *linda_client*
- linda_client/2 - *linda_client*
- linda_timeout/2 - *linda_client*
- line_to_bytes/2 - *reader*
- line_to_bytes/3 - *reader*
- line_to_chars/2 - *reader*

- line_to_chars/3 - *reader*
- line_to_codes/2 - *reader*
- line_to_codes/3 - *reader*
- linear_regression/4 - *numberlistp*
- lint/0 - *packs*, *registries*
- lint/1 - *packs*, *registries*
- lint/2 - *packs*
- list/0 - *registries*
- list_buckets/2 - *s3_protocol*
- list_buckets/3 - *s3_session(HTTPTransport)*
- list_buckets_response/2 - *s3_xml*
- list_objects_v2/4 - *s3_protocol*
- list_objects_v2/5 - *s3_session(HTTPTransport)*
- list_objects_v2_response/2 - *s3_xml*
- list_to_array/2 - *java_utils_protocol*
- listing/0 - *listing*
- listing/1 - *listing*
- literal/2 - *sat_protocol*
- llen/3 - *redis*
- load/1 - *mime_types*, *tzif_protocol*
- load/2 - *http_cookie_jar*, *mime_types*, *tzif_protocol*
- load_config/1 - *subprocess_mutation_hook*
- load_coverage_config/1 - *subprocess_coverage_hook*
- load_dut1_override/1 - *time_scales_data*, *time_scales_protocol*
- load_leap_seconds_override/1 - *time_scales_data*, *time_scales_protocol*
- load_program/1 - *datalog_protocol*
- loaded_file/1 - *logtalk*
- loaded_file_property/2 - *logtalk*, *modules_diagram_support*
- loaded_files_topological_sort/1 - *logtalk*
- loaded_files_topological_sort/2 - *logtalk*
- loaded_pack/3 - *packs*
- loaded_pack_dependency/6 - *packs*
- loaded_pack_file/4 - *packs*
- loader_file/1 - *application_protocol*
- local_abbreviation/2 - *tzif_protocol*
- local_abbreviation/3 - *tzif_protocol*
- local_abbreviation/4 - *tzif_protocol*
- local_abbreviation_reified/2 - *tzif_protocol*
- local_abbreviation_reified/3 - *tzif_protocol*
- local_abbreviation_reified/4 - *tzif_protocol*
- local_abbreviation_with_resolution/3 - *tzif_protocol*
- local_abbreviation_with_resolution/4 - *tzif_protocol*
- local_abbreviation_with_resolution/5 - *tzif_protocol*
- local_daylight_saving_time/2 - *tzif_protocol*
- local_daylight_saving_time/3 - *tzif_protocol*
- local_daylight_saving_time/4 - *tzif_protocol*
- local_daylight_saving_time_reified/2 - *tzif_protocol*
- local_daylight_saving_time_reified/3 - *tzif_protocol*
- local_daylight_saving_time_reified/4 - *tzif_protocol*

- `local_daylight_saving_time_with_resolution/3` - *tzif_protocol*
- `local_daylight_saving_time_with_resolution/4` - *tzif_protocol*
- `local_daylight_saving_time_with_resolution/5` - *tzif_protocol*
- `local_decrypt/3` - *paseto_protocol*
- `local_decrypt/5` - *paseto_protocol*
- `local_encrypt/3` - *paseto_protocol*
- `local_encrypt/5` - *paseto_protocol*
- `local_key/1` - *paseto_protocol*
- `local_offset/2` - *tzif_protocol*
- `local_offset/3` - *tzif_protocol*
- `local_offset/4` - *tzif_protocol*
- `local_offset_reified/2` - *tzif_protocol*
- `local_offset_reified/3` - *tzif_protocol*
- `local_offset_reified/4` - *tzif_protocol*
- `local_offset_with_resolution/3` - *tzif_protocol*
- `local_offset_with_resolution/4` - *tzif_protocol*
- `local_offset_with_resolution/5` - *tzif_protocol*
- `local_tangent_plane/3` - *crs_projections_protocol*
- `local_tangent_plane_inverse/3` - *crs_projections_protocol*
- `local_time_type/2` - *tzif_protocol*
- `local_time_type/3` - *tzif_protocol*
- `local_time_type/4` - *tzif_protocol*
- `local_time_type_reified/2` - *tzif_protocol*
- `local_time_type_reified/3` - *tzif_protocol*
- `local_time_type_reified/4` - *tzif_protocol*
- `local_time_type_with_resolution/3` - *tzif_protocol*
- `local_time_type_with_resolution/4` - *tzif_protocol*
- `local_time_type_with_resolution/5` - *tzif_protocol*
- `local_to_utc/3` - *datep*
- `local_to_utc_tz/3` - *dates_tz_protocol*
- `local_to_utc_tz_with_resolution/4` - *dates_tz_protocol*
- `location/2` - *gravatar*
- `locked_pairs/2` - *ranked_pairs*
- `log/3` - *debuggerp*
- `log_event/2` - *loggingp*
- `log_file/2` - *loggingp*
- `logging/1` - *loggingp*
- `logging/3` - *debuggerp*
- `logistic/3` - *sampling_protocol*
- `logistic_normal/3` - *multivariate_distributions_protocol*
- `logistic_normal/4` - *multivariate_distributions_protocol*
- `logistic_normal_samples/4` - *multivariate_distributions_protocol*
- `logistic_normal_samples/5` - *multivariate_distributions_protocol*
- `lognormal/3` - *sampling_protocol*
- `logout_url/4` - *open_id*, *open_id_logout*
- `logseries/2` - *sampling_protocol*
- `logtalk_packs/0` - *packs_common*
- `logtalk_packs/1` - *packs_common*
- `long_flags/1` - *command_line_option*

- longest_common_increasing_subsequence/3 - *subsequences_protocol*
- longest_common_subsequence/3 - *string_distance(Representation)*, *subsequences_protocol*
- longest_common_subsequence_length/3 - *string_distance(Representation)*
- longest_common_substring/3 - *string_distance(Representation)*
- longest_decreasing_subsequence/2 - *subsequences_protocol*
- longest_increasing_subsequence/2 - *subsequences_protocol*
- longest_prefix/4 - *trie(Representation)*
- longest_repeating_subsequence/2 - *subsequences_protocol*
- lookup/2 - *dictionaryp*
- lookup/3 - *dictionaryp*, *lru_cache_protocol*, *trie(Representation)*
- lookup/4 - *dictionaryp*
- lookup_in/3 - *nested_dictionary_protocol*
- lookup_prefix/4 - *trie(Representation)*
- lower_case/2 - *text_normalizer_protocol*
- lower_triangular_part/2 - *linear_algebra_protocol*
- lower_triangular_part/3 - *linear_algebra_protocol*
- lower_upper/2 - *characterp*
- lpop/3 - *redis*
- lpush/4 - *redis*
- lrange/5 - *redis*
- lrem/5 - *redis*
- ltrim/5 - *redis*
- lucas/2 - *natural*
- magic/2 - *magic*
- magicise/4 - *magic*
- mahalanobis_distance/4 - *multivariate_distributions_protocol*
- mahalanobis_distance/5 - *multivariate_distributions_protocol*
- make_directory/1 - *osp*
- make_directory_path/1 - *osp*
- make_set/3 - *union_find_protocol*
- man/1 - *help*
- manhattan_distance/3 - *numberlistp*
- manhattan_norm/2 - *numberlistp*
- map/2 - *deque_protocol*, *dictionaryp*, *expected(Expected)*, *metap*, *optional(Optional)*, *queuep*, *validation(Validation)*
- map/3 - *deque_protocol*, *dictionaryp*, *metap*, *pairs*, *queuep*, *validated*
- map/4 - *metap*, *validated*
- map/5 - *metap*
- map/6 - *metap*
- map/7 - *metap*
- map/8 - *metap*
- map_both/3 - *expected(Expected)*, *validation(Validation)*
- map_catching/2 - *expected(Expected)*, *validation(Validation)*
- map_element/2 - *java_utils_protocol*
- map_id/2 - *ccsds_tc_services*
- map_invalid/2 - *validation(Validation)*
- map_or_else/3 - *expected(Expected)*, *optional(Optional)*, *validation(Validation)*
- map_reduce/5 - *metap*
- map_unexpected/2 - *expected(Expected)*
- mask32/1 - *hash_common_32*

- mask64/1 - *hash_common_64*
- master_channel_frame_count/2 - *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- match/1 - *multiplayer_ranking_dataset_protocol*
- materialize/0 - *datalog_protocol*
- matrix_column/3 - *linear_algebra_protocol*
- matrix_column_means/2 - *linear_algebra_protocol*
- matrix_diagonal/2 - *linear_algebra_protocol*
- matrix_diagonal/3 - *linear_algebra_protocol*
- matrix_matrix_product/3 - *linear_algebra_protocol*
- matrix_rank/2 - *linear_algebra_protocol*
- matrix_rank/3 - *linear_algebra_protocol*
- matrix_row/3 - *linear_algebra_protocol*
- matrix_row_means/2 - *linear_algebra_protocol*
- matrix_trace/2 - *linear_algebra_protocol*
- matrix_value/4 - *linear_algebra_protocol*
- matrix_vector_product/3 - *linear_algebra_protocol*
- max/2 - *listp*, *numberlistp*, *statisticsp*
- max/3 - *dictionaryp*
- max_clauses/1 - *metagol*
- max_flow/5 - *max_flow_protocol*
- max_inv_preds/1 - *metagol*
- max_path/5 - *graph_protocol*
- max_size/1 - *arbitrary*
- max_tree/3 - *weighted_undirected_graph(Dictionary)*
- maximal_cliques/2 - *undirected_graph_common*
- maximum_cliques/2 - *undirected_graph_common*
- maybe/0 - *random_protocol*
- maybe/1 - *random_protocol*
- maybe/2 - *random_protocol*
- maybe_call/1 - *random_protocol*
- maybe_call/2 - *random_protocol*
- mean_center/2 - *geospatial_protocol*
- mean_deviation/2 - *statisticsp*
- mean_squared_error/3 - *statisticsp*
- measurement/4 - *pairwise_measurement_dataset_protocol*
- media_type/2 - *http_multipart*
- media_types/1 - *http_body_codec_protocol*
- median/2 - *numberlistp*, *statisticsp*
- median_deviation/2 - *statisticsp*
- meets/2 - *interval_protocol*
- member/2 - *interval_relation_set_protocol*, *listp*, *random_protocol*, *setp*
- memberchk/2 - *listp*, *setp*, *varlistp*
- merge/3 - *heap_protocol*
- merged_options/2 - *http_websocket_service_loop_options*, *http_websocket_session_read_options*, *http_websocket_session_write_options*
- message/3 - *http_websocket_messages(TextRepresentation)*
- message_body/2 - *amqp*
- message_delivery_tag/2 - *amqp*
- message_exchange/2 - *amqp*

- message_hook/4 - *logtalk*
- message_prefix_file/6 - *logtalk*
- message_prefix_stream/4 - *logtalk*
- message_properties/2 - *amqp*
- message_property/3 - *amqp*
- message_routing_key/2 - *amqp*
- message_tokens//2 - *logtalk*
- met_by/2 - *interval_protocol*
- meta/1 - *command_line_option*
- meta_type/3 - *type*
- metaphone/2 - *string_distance(Representation)*
- metaphone_match/2 - *string_distance(Representation)*
- metarule/6 - *metagol*
- metarule_next_id/1 - *metagol*
- method/1 - *http_request_protocol*
- method/2 - *http_core*, *json_rpc*
- method_not_found/2 - *json_rpc*
- mget/3 - *memcached*, *redis*
- mibenum/1 - *character_set_protocol*
- midpoint/3 - *geospatial_protocol*
- milliseconds/2 - *ccsds_cds(DaySegmentOctets,SubmillisecondOctets,Epoch)*
- min/2 - *listp*, *numberlistp*, *statisticsp*
- min/3 - *dictionaryp*
- min_clauses/1 - *metagol*
- min_cost_flow/4 - *min_cost_flow_protocol*
- min_cost_flow/5 - *min_cost_flow_protocol*
- min_cost_max_flow/6 - *min_cost_flow_protocol*
- min_cost_max_flow/7 - *min_cost_flow_protocol*
- min_cut/5 - *max_flow_protocol*
- min_cut_value/4 - *gomory_hu(Dictionary)*
- min_cuts/5 - *max_flow_protocol*
- min_distances/3 - *graph_protocol*
- min_max/3 - *numberlistp*, *statisticsp*
- min_max_normalization/2 - *statisticsp*
- min_path/5 - *graph_protocol*
- min_path_bellman_ford/5 - *weighted_graph_protocol*
- min_paths/3 - *weighted_directed_graph(Dictionary)*
- min_predecessors/3 - *graph_protocol*
- min_tree/3 - *weighted_undirected_graph(Dictionary)*
- mine/2 - *pattern_miner_protocol*
- mine/3 - *association_rule_miner_protocol*, *pattern_miner_protocol*
- mine/4 - *association_rule_miner_protocol*
- minimum_enclosing_circle/3 - *geospatial_protocol*
- minizinc_statement/2 - *minizinc_problem_protocol*
- minkowski_distance/4 - *numberlistp*
- model/2 - *asp_protocol*, *minizinc_protocol*
- model_binding/2 - *omt_protocol*, *smt_protocol*
- model_value/3 - *omt_protocol*, *smt_protocol*
- modes/2 - *numberlistp*, *statisticsp*

- module_property/2 - *modules_diagram_support*
- monitor/1 - *event_registryp*
- monitor/4 - *event_registryp*
- monitor_activated/0 - *monitorp*
- monitored/1 - *event_registryp*
- monitors/1 - *event_registryp*
- month/2 - *ccsds_ccs(CalendarVariant,FractionOctets)*
- month_weekday_date/5 - *datep*
- mset/3 - *redis*
- msort/2 - *listp*
- msort/3 - *listp*
- mul32/3 - *hash_common_32*
- mul64/3 - *hash_common_64*
- multi_source_max_flow/5 - *max_flow_protocol*
- multinomial/2 - *natural*
- multinomial/3 - *multivariate_distributions_protocol*
- multinomial_density/4 - *multivariate_distributions_protocol*
- multinomial_log_density/4 - *multivariate_distributions_protocol*
- multinomial_quantile/4 - *multivariate_distributions_protocol*
- multinomial_samples/4 - *multivariate_distributions_protocol*
- multiplayer_dataset_connected_components/2 - *ranking_dataset_common*
- multiplayer_dataset_declared_items/2 - *ranking_dataset_common*
- multiplayer_dataset_declared_matches/2 - *ranking_dataset_common*
- multiplayer_dataset_events/2 - *ranking_dataset_common*
- multiplayer_dataset_items/2 - *ranking_dataset_common*
- multiplayer_dataset_match_teams/3 - *ranking_dataset_common*
- multiplayer_dataset_matches/2 - *ranking_dataset_common*
- multiplayer_dataset_summary/2 - *ranking_dataset_common*
- multiset/3 - *multisets_protocol*
- multiset/4 - *multisets_protocol*
- multiset_index/4 - *multisets_protocol*
- multiset_index/5 - *multisets_protocol*
- multisets/3 - *multisets_protocol*
- multisets/4 - *multisets_protocol*
- multivariate_normal/3 - *multivariate_distributions_protocol*
- multivariate_normal/4 - *multivariate_distributions_protocol*
- multivariate_normal_density/4 - *multivariate_distributions_protocol*
- multivariate_normal_density/5 - *multivariate_distributions_protocol*
- multivariate_normal_log_density/4 - *multivariate_distributions_protocol*
- multivariate_normal_log_density/5 - *multivariate_distributions_protocol*
- multivariate_normal_samples/4 - *multivariate_distributions_protocol*
- multivariate_normal_samples/5 - *multivariate_distributions_protocol*
- multivariate_t/4 - *multivariate_distributions_protocol*
- multivariate_t/5 - *multivariate_distributions_protocol*
- multivariate_t_density/5 - *multivariate_distributions_protocol*
- multivariate_t_density/6 - *multivariate_distributions_protocol*
- multivariate_t_log_density/5 - *multivariate_distributions_protocol*
- multivariate_t_log_density/6 - *multivariate_distributions_protocol*
- multivariate_t_samples/5 - *multivariate_distributions_protocol*

- `multivariate_t_samples/6` - *multivariate_distributions_protocol*
- `must_precede/3` - *stn_protocol*
- `must_precede_or_equal/3` - *stn_protocol*
- `mutate/2` - *genetic_algorithm_problem_protocol*
- `mutation/2` - *mutator_protocol*
- `mutation/3` - *mutations*, *mutations_store*
- `mutation_rate/4` - *genetic_algorithm_problem_protocol*
- `n_grams/3` - *n_grams_protocol*
- `n_grams/4` - *n_grams_protocol*
- `nack/3` - *stomp*
- `name/1` - *application_protocol*, *character_set_protocol*, *command_line_option*, *pack_protocol*, *registry_protocol*
- `name_of_day/3` - *datep*
- `name_of_month/3` - *datep*
- `named_entity/2` - *text_normalization_profile_protocol*
- `nan_kind/2` - *ieee_754_fields_protocol*
- `nan_payload/2` - *ieee_754_fields_protocol*
- `nan_representation/1` - *ieee_754_protocol*
- `natural//1` - *number_grammars(Format)*
- `nearest_coordinate/5` - *geospatial_protocol*
- `nearest_point_on_polyline/4` - *geospatial_protocol*
- `nearest_point_on_segment/4` - *geospatial_protocol*
- `neighbor_state/2` - *simulated_annealing_protocol*, *tabu_search_problem_protocol*
- `neighbor_state/3` - *simulated_annealing_protocol*, *tabu_search_problem_protocol*
- `neighbors/2` - *geohash_protocol*, *tabu_search_problem_protocol*
- `neighbors/3` - *graph_protocol*
- `neighbors_bits/3` - *geohash_protocol*
- `neighbors_int/3` - *geohash_protocol*
- `network/1` - *interval_constraint_network_protocol*
- `new/1` - *atms_protocol*, *graph_protocol*, *java_access_protocol*, *nested_dictionary_protocol*, *ripple_down_rules_common_protocol*, *streamvars*, *termp*, *trie(Representation)*
- `new/2` - *atms_protocol*, *flow_network_protocol*, *graph_protocol*, *interval_constraint_network_protocol*, *java_access_protocol*, *lru_cache_protocol*, *ripple_down_rules_common_protocol*, *stn_protocol*, *streamvars*, *union_find_protocol*
- `new/3` - *flow_network_protocol*, *graph_protocol*, *interval_protocol*
- `new_hash_state/1` - *hash_state_protocol*
- `new_line//0` - *blank_grammars(Format)*
- `new_lines//0` - *blank_grammars(Format)*
- `new_matrix/4` - *linear_algebra_protocol*
- `new_problem/1` - *linear_programming_protocol*
- `new_vector/3` - *linear_algebra_protocol*
- `new_vector_like/2` - *linear_algebra_protocol*
- `next/2` - *zipperp*
- `next/3` - *zipperp*
- `next/4` - *dictionaryp*, *http_sse_service_handler_protocol*
- `next_arrangement/3` - *arrangements_protocol*
- `next_cartesian_tuple/3` - *cartesian_products_protocol*
- `next_combination/3` - *combinations_protocol*
- `next_derangement/3` - *derangements_protocol*
- `next_distinct_partition/3` - *integer_partitions_protocol*
- `next_distinct_partition/4` - *integer_partitions_protocol*

- next_multiset/3 - *multisets_protocol*
- next_partition/3 - *integer_partitions_protocol*, *partitions_protocol*
- next_partition/4 - *integer_partitions_protocol*, *partitions_protocol*
- next_permutation/2 - *permutations_protocol*
- nextto/3 - *listp*, *varlistp*
- no_content_response/2 - *rest*
- node/2 - *atms_protocol*
- node/3 - *json_graph_data_protocol*
- node/7 - *graph_language_protocol*
- node_capacity_max_flow/6 - *max_flow_protocol*
- nodebug/0 - *debuggerp*
- nodes/1 - *ant_colony_problem_protocol*
- nodes/2 - *atms_protocol*, *interval_constraint_network_protocol*
- nodes/3 - *json_path_protocol*
- nogood/2 - *atms_protocol*
- nolog/3 - *debuggerp*
- nologall/0 - *debuggerp*
- non_blank//1 - *blank_grammars(Format)*
- non_blanks//1 - *blank_grammars(Format)*
- nonempty_subsequence/2 - *subsequences_protocol*
- nonempty_subsequences/2 - *subsequences_protocol*
- normal/3 - *sampling_protocol*, *univariate_distributions_protocol*
- normal_density/4 - *univariate_distributions_protocol*
- normal_distribution/4 - *univariate_distributions_protocol*
- normal_element/2 - *html*
- normal_log_density/4 - *univariate_distributions_protocol*
- normal_quantile/4 - *univariate_distributions_protocol*
- normal_samples/4 - *univariate_distributions_protocol*
- normalize/2 - *interval_relation_set_protocol*, *url(Representation)*
- normalize_cookie_attributes/2 - *http_cookies(Representation)*
- normalize_coordinate/2 - *geospatial_protocol*
- normalize_date_time/2 - *datep*
- normalize_polygon_orientation/3 - *geospatial_protocol*
- normalize_range/2 - *numberlistp*
- normalize_range/4 - *numberlistp*
- normalize_scalar/2 - *numberlistp*
- normalize_unicode/3 - *text_normalizer_protocol*
- normalize_unit/2 - *numberlistp*
- normalize_vector/2 - *linear_algebra_protocol*
- normalize_vector/3 - *linear_algebra_protocol*
- normalize_whitespace/2 - *text_normalizer_protocol*
- normalize_whitespace/3 - *text_normalizer_protocol*
- nospy/1 - *debuggerp*
- nospy/3 - *debuggerp*
- nospy/4 - *debuggerp*
- nospyall/0 - *debuggerp*
- not64/2 - *hash_common_64*
- note/2 - *registry_protocol*
- note/3 - *pack_protocol*

- notification/2 - *json_rpc*
- notification/3 - *json_rpc*
- notify/1 - *mcp_server*, *mcp_server_spec_protocol*, *mcp_server_transport_protocol*
- notrace/0 - *debuggerp*
- now/3 - *timep*
- nth0/3 - *listp*, *varlistp*
- nth0/4 - *listp*, *varlistp*
- nth1/3 - *listp*, *varlistp*
- nth1/4 - *listp*, *varlistp*
- nth_arrangement/4 - *arrangements_protocol*
- nth_arrangement/5 - *arrangements_protocol*
- nth_cartesian_tuple/3 - *cartesian_products_protocol*
- nth_cartesian_tuple/4 - *cartesian_products_protocol*
- nth_combination/4 - *combinations_protocol*
- nth_combination/5 - *combinations_protocol*
- nth_derangement/3 - *derangements_protocol*
- nth_derangement/4 - *derangements_protocol*
- nth_distinct_arrangement/4 - *arrangements_protocol*
- nth_distinct_cartesian_tuple/3 - *cartesian_products_protocol*
- nth_distinct_combination/4 - *combinations_protocol*
- nth_distinct_derangement/3 - *derangements_protocol*
- nth_distinct_multiset/4 - *multisets_protocol*
- nth_distinct_partition/3 - *integer_partitions_protocol*, *partitions_protocol*
- nth_distinct_partition/4 - *integer_partitions_protocol*, *partitions_protocol*
- nth_distinct_partition/5 - *integer_partitions_protocol*, *partitions_protocol*
- nth_distinct_permutation/3 - *permutations_protocol*
- nth_multiset/4 - *multisets_protocol*
- nth_multiset/5 - *multisets_protocol*
- nth_partition/3 - *integer_partitions_protocol*, *partitions_protocol*
- nth_partition/4 - *integer_partitions_protocol*, *partitions_protocol*
- nth_partition/5 - *integer_partitions_protocol*, *partitions_protocol*
- nth_permutation/3 - *permutations_protocol*
- nth_permutation/4 - *permutations_protocol*
- null/1 - *java_utils_protocol*
- null_device_path/1 - *osp*
- null_space/2 - *linear_algebra_protocol*
- null_space/3 - *linear_algebra_protocol*
- number//1 - *number_grammars(Format)*
- number_of_edges/2 - *graph_protocol*
- number_of_tests/1 - *lgtunit*
- number_of_vertices/2 - *graph_protocol*
- number_string/2 - *string(Representation)*
- numbervars/1 - *termp*
- numbervars/3 - *termp*
- nysiis/2 - *string_distance(Representation)*
- nysiis_match/2 - *string_distance(Representation)*
- objective/2 - *local_optimization_problem_protocol*
- objective/3 - *omt_protocol*
- objective/4 - *linear_programming_protocol*

- objective_value/2 - *linear_programming_protocol*
- observation/2 - *time_series_dataset_protocol*
- occurrences/2 - *listp*
- occurrences/3 - *listp*
- occurs/2 - *termp*
- ocf/2 - *ccsds_frame_protocol*, *ccsds_frames*
- ocf_flag/2 - *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- of/2 - *optional*
- of_expected/2 - *expected*
- of_invalid/2 - *validation*
- of_invalids/2 - *validation*
- of_unexpected/2 - *expected*
- of_valid/2 - *validation*
- offset/2 - *tzif_protocol*
- offset/3 - *time_scales_protocol*, *tzif_protocol*
- offset/4 - *tzif_protocol*
- one_or_more//0 - *sequence_grammars*
- one_or_more//1 - *sequence_grammars*
- one_or_more//2 - *sequence_grammars*
- opcode/2 - *http_websocket_frames*
- open/1 - *http_client_session(HTTPTransport)*, *http_cookie_jar*, *http_server_core_session*, *http_websocket_service_registry*, *s3_session(HTTPTransport)*
- open/2 - *http_client_session(HTTPTransport)*, *http_cookie_jar*, *http_server_core_session*, *http_sse*, *http_websocket*, *s3_session(HTTPTransport)*
- open/3 - *http_client_digest_session(HTTPTransport)*, *http_server*, *http_sse*, *http_websocket*
- open/4 - *http_client_digest_session(HTTPTransport)*, *http_server*, *http_websocket_client_service(HTTPTransport)*
- open/5 - *http_websocket_client_service(HTTPTransport)*
- open/8 - *smtp_process_transport*
- open_api_bad_request_response/1 - *http_parameters*
- open_api_parameters/2 - *http_parameters*
- open_api_request_body/3 - *http_parameters*
- open_connection/4 - *http_transport_protocol*
- open_connection_pool/4 - *http_transport_protocol*
- open_listener/4 - *http_transport_protocol*
- open_session/4 - *http_sse*, *http_websocket*
- open_session/5 - *http_sse*, *http_websocket*
- open_sse/4 - *http_client*
- open_websocket/4 - *http_client*
- operating_system_machine/1 - *osp*
- operating_system_name/1 - *osp*
- operating_system_release/1 - *osp*
- operating_system_type/1 - *osp*
- operation/3 - *open_api*
- operation/8 - *open_ai_catalog*
- operation_count/1 - *open_ai*, *open_ai_catalog*
- operation_properties/2 - *open_ai_catalog*
- operation_url/4 - *open_ai_client*
- operations/1 - *http_router*, *open_api_provider_protocol*
- option/2 - *options_protocol*

- option/3 - *options_protocol*
- or/2 - *expected(Expected)*, *optional(Optional)*, *validation(Validation)*
- or64/3 - *hash_common_64*
- or_else/2 - *expected(Expected)*, *optional(Optional)*, *validation(Validation)*
- or_else_call/2 - *expected(Expected)*, *optional(Optional)*, *validation(Validation)*
- or_else_fail/1 - *expected(Expected)*, *optional(Optional)*, *validation(Validation)*
- or_else_get/2 - *expected(Expected)*, *optional(Optional)*, *validation(Validation)*
- or_else_throw/1 - *expected(Expected)*, *validation(Validation)*
- or_else_throw/2 - *expected(Expected)*, *optional(Optional)*, *validation(Validation)*
- order/1 - *ieee_754_fields_protocol*, *ieee_754_protocol*
- originator/1 - *application_protocol*
- orphaned/0 - *packs*
- orphaned/2 - *packs*
- out/1 - *linda_client*
- out/2 - *linda_client*
- out_degree/3 - *directed_graph_protocol*
- outdated/0 - *packs*
- outdated/1 - *packs*
- outdated/2 - *packs*
- outdated/4 - *packs*
- outdated/5 - *packs*
- outer_product/3 - *linear_algebra_protocol*
- output_file_name/2 - *graph_language_protocol*
- output_schema/2 - *mcp_tool_protocol*
- overlapped_by/2 - *interval_protocol*
- overlaps/2 - *interval_protocol*
- pack_dependency/6 - *packs*
- pack_metadata/4 - *packs*
- pack_object/3 - *packs*
- pack_property/4 - *packs*
- package/1 - *application_protocol*
- packet_order_flag/2 - *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- packet_type/2 - *mqtt*
- packetize_aos_frames/7 - *ccsds_packetization*
- packetize_aos_frames/8 - *ccsds_packetization*
- packetize_aos_packets/6 - *ccsds_packetization*
- packetize_aos_packets/7 - *ccsds_packetization*
- packetize_tm_frames/7 - *ccsds_packetization*
- packetize_tm_frames/8 - *ccsds_packetization*
- packetize_tm_packets/6 - *ccsds_packetization*
- packetize_tm_packets/7 - *ccsds_packetization*
- packets/2 - *ccsds_packet_services*
- pad_md/4 - *hash_common_32*
- pad_md_tail/3 - *hash_common_64*
- pad_md_tail/5 - *hash_common_32*
- page_fragment_reply/4 - *http_htmx*
- page_fragment_reply/5 - *http_htmx*
- pairwise_dataset_connected_components/2 - *ranking_dataset_common*
- pairwise_dataset_declared_items/2 - *ranking_dataset_common*

- pairwise_dataset_items/2 - *ranking_dataset_common*
- pairwise_dataset_matchups/2 - *ranking_dataset_common*
- pairwise_dataset_preferences/2 - *ranking_dataset_common*
- pairwise_dataset_summary/2 - *ranking_dataset_common*
- pairwise_dataset_win_totals/2 - *ranking_dataset_common*
- pairwise_measurement_dataset_connected_components/2 - *ranking_dataset_common*
- pairwise_measurement_dataset_declared_items/2 - *ranking_dataset_common*
- pairwise_measurement_dataset_items/2 - *ranking_dataset_common*
- pairwise_measurement_dataset_measurements/2 - *ranking_dataset_common*
- pairwise_measurement_dataset_summary/2 - *ranking_dataset_common*
- parameter/3 - *http_parameters*
- parameters/3 - *http_parameters*
- params/2 - *json_rpc*
- parent/1 - *proto_hierarchyp*
- parent/2 - *geohash_protocol*
- parenthesis/2 - *characterp*
- parents/1 - *proto_hierarchyp*
- parse/2 - *asp_symbols*, *avro*, *base32*, *base58*, *base64*, *base64url*, *base64url_no_padding*, *base85*, *bson(StringRepresentation)*, *cbor(StringRepresentation)*, *ccsds_frame_protocol*, *ccsds_packets(SecondaryHeaderLength)*, *ccsds_time_code_protocol*, *crontab(Representation)*, *geojson_protocol*, *gpx_protocol*, *http_websocket_frames*, *ieee_754_protocol*, *json_graph_protocol*, *json_ld_protocol*, *json_lines_protocol*, *json_path_protocol*, *json_pointer_protocol*, *json_protocol*, *json_schema_protocol*, *message_pack(StringRepresentation)*, *nmea_protocol*, *open_api*, *protobuf*, *tle_orbits_protocol*, *toml_protocol*, *toon_protocol*, *url(Representation)*, *wkt_wkb_protocol*, *xml*, *yaml_protocol*
- parse/3 - *avro*, *ccsds_time_fields*, *crontab(Representation)*, *nmea_protocol*, *protobuf*, *url(Representation)*, *xml*
- parse/4 - *command_line_options*, *http_multipart*
- parse/5 - *command_line_options*
- parse_all/2 - *yaml_protocol*
- parse_authentication_info/2 - *http_digest*
- parse_authorization/2 - *http_authenticate*, *http_digest*, *http_oauth*
- parse_body/4 - *http_core*
- parse_challenge/2 - *http_authenticate*, *http_digest*, *http_oauth*
- parse_cookie/2 - *http_cookies(Representation)*
- parse_domain/2 - *pddl*
- parse_domain/3 - *pddl*
- parse_error/1 - *json_rpc*
- parse_fragment/2 - *json_pointer_protocol*
- parse_frame/3 - *ccsds_link_profiles*
- parse_frames/3 - *ccsds_link_profiles*
- parse_headers/2 - *http_core*
- parse_lines/4 - *tle_orbits_protocol*
- parse_problem/2 - *pddl*
- parse_problem/3 - *pddl*
- parse_relative/2 - *json_pointer_protocol*
- parse_request/2 - *http_core*
- parse_request_line/2 - *http_core*
- parse_response/2 - *http_core*
- parse_set_cookie/4 - *http_cookies(Representation)*
- parse_status_line/2 - *http_core*
- part/4 - *http_multipart*
- part_body/2 - *http_multipart*

- part_headers/2 - *http_multipart*
- part_properties/2 - *http_multipart*
- partial_map/4 - *rbtree*
- partition/2 - *integer_partitions_protocol*, *partitions_protocol*
- partition/3 - *either*, *integer_partitions_protocol*, *partitions_protocol*, *validated*
- partition/4 - *integer_partitions_protocol*, *metap*, *partitions_protocol*
- partition/5 - *listp*
- partition/6 - *metap*
- partition_index/3 - *integer_partitions_protocol*, *partitions_protocol*
- partition_index/4 - *integer_partitions_protocol*, *partitions_protocol*
- partition_index/5 - *integer_partitions_protocol*, *partitions_protocol*
- partition_number/2 - *natural*
- partitions/2 - *integer_partitions_protocol*, *partitions_protocol*
- partitions/3 - *integer_partitions_protocol*, *partitions_protocol*
- partitions/4 - *integer_partitions_protocol*, *partitions_protocol*
- parts/2 - *http_multipart*
- password_hash/4 - *crypto*
- password_hash_atom/2 - *crypto*
- password_hash_needs_rehash/3 - *crypto*
- patch/4 - *http_client*
- patch/5 - *http_client*, *http_client_digest_session(HTTPTransport)*, *http_client_session(HTTPTransport)*
- path/3 - *graph_protocol*
- path_concat/3 - *osp*
- path_consistency/2 - *interval_constraint_network_protocol*
- path_parameter/3 - *rest*
- paths/3 - *json_path_protocol*
- pattern_miner_options/2 - *pattern_miner_protocol*
- payload/2 - *http_websocket_frames*, *http_websocket_messages(TextRepresentation)*
- payload_hash/2 - *s3_signing*
- pbkdf2/6 - *crypto*
- peek_algorithm/2 - *jwt*
- peek_back/2 - *deque_protocol*
- peek_front/2 - *deque_protocol*
- peek_key_id/2 - *jwt*, *paseto*
- pending_data/2 - *ccsds_packet_services*
- pending_fragments/2 - *ccsds_link_profiles*, *ccsds_packet_services*, *ccsds_tc_services*
- pending_packets/2 - *ccsds_packetization*
- percentile/3 - *statisticsp*
- period/1 - *temporal_pairwise_ranking_dataset_protocol*
- permutation/2 - *listp*, *permutations_protocol*, *random_protocol*, *varlistp*
- permutation/3 - *permutations_protocol*
- permutation_index/3 - *permutations_protocol*
- permutation_index/4 - *permutations_protocol*
- permutations/2 - *permutations_protocol*
- permutations/3 - *permutations_protocol*
- persist/3 - *redis*
- pid/1 - *osp*
- pin/0 - *packs_common*
- pin/1 - *packs_common*

- `ping/2` - *mqtt*
- `pinned/1` - *packs_common*
- `plus/3` - *integer*
- `point_in_polygon/2` - *geospatial_protocol*
- `point_to_polyline_distance/3` - *geospatial_protocol*
- `poisson/2` - *sampling_protocol*
- `polygon_area/2` - *geospatial_protocol*
- `polygon_bounding_box/2` - *geospatial_protocol*
- `polygon_centroid/2` - *geospatial_protocol*
- `polygon_covering/4` - *geohash_protocol*
- `polygon_orientation/2` - *geospatial_protocol*
- `polygon_perimeter/2` - *geospatial_protocol*
- `polygon_perimeter/3` - *geospatial_protocol*
- `polygons_intersect/2` - *geospatial_protocol*
- `polyline_covering/4` - *geohash_protocol*
- `polyline_length/2` - *geospatial_protocol*
- `polyline_length/3` - *geospatial_protocol*
- `polyline_resample/3` - *geospatial_protocol*
- `polyline_simplify/3` - *geospatial_protocol*
- `polyline_split_at_distance/4` - *geospatial_protocol*
- `pop_back/3` - *deque_protocol*
- `pop_front/3` - *deque_protocol*
- `port/5` - *ports_profiler*
- `portray_clause/1` - *listing*
- `position_bounds/1` - *differential_evolution_problem_protocol*, *local_optimization_problem_protocol*, *particle_swarm_optimization_protocol*
- `possible/4` - *interval_constraint_network_protocol*
- `post/4` - *http_client*
- `post/5` - *http_client*, *http_client_digest_session(HTTPTransport)*, *http_client_session(HTTPTransport)*
- `postorder/2` - *bintree*
- `power/2` - *sampling_protocol*
- `power_sequence/4` - *integer*
- `power_set/2` - *subsequences_protocol*
- `powerset/2` - *setp*
- `pp/1` - *xml*
- `pprint/1` - *metagol*
- `precision/1` - *ieee_754_fields_protocol*, *ieee_754_protocol*
- `precision/3` - *geohash_protocol*
- `predicate/1` - *caller_diagram(Format)*
- `predicate/2` - *caller_diagram(Format)*, *dead_code_scanner*, *mutation_testing*
- `predicate/3` - *dead_code_scanner*, *mutation_testing*
- `predicate_info_pair_score_hook/4` - *doc_metric*
- `predicate_info_score_hook/3` - *doc_metric*
- `predicate_mode_score_hook/3` - *doc_metric*
- `predicate_mode_score_hook/5` - *doc_metric*
- `predicate_mutants/3` - *mutation_testing*
- `predicate_mutants/4` - *mutation_testing*
- `predicate_stratum/3` - *datalog_protocol*
- `predicates/2` - *dead_code_scanner*

- predicates/3 - *dead_code_scanner*
- predict/3 - *anomaly_detector_protocol*, *classifier_protocol*, *regressor_protocol*
- predict/4 - *anomaly_detector_protocol*, *knn_classifier*, *nearest_centroid_classifier*
- predict_distribution/3 - *bayesian_ridge_regression*, *gaussian_process_regression*
- predict_probabilities/3 - *probabilistic_classifier_protocol*
- predict_probabilities/4 - *knn_classifier*, *nearest_centroid_classifier*
- predict_scores/3 - *lda_classifier*, *qda_classifier*
- preference/3 - *pairwise_ranking_dataset_protocol*
- preferred_mime_name/1 - *character_set_protocol*
- prefix/0 - *packs_common*
- prefix/1 - *packs_common*
- prefix/2 - *listp*, *typeid_protocol*, *varlistp*
- prefix/3 - *listp*
- prefix_data/2 - *ccsds_packet_services*
- prefixes/3 - *trie(Representation)*
- preflight_response/3 - *http_cors*
- preorder/2 - *bintree*
- prepare/2 - *mcp_server_spec_protocol*, *mcp_server_streamable_http_transport*
- prepare_key/2 - *block_cipher_prepared_key_protocol*
- prepend/3 - *memcached*
- presigned_get_object/4 - *s3_protocol*
- presigned_get_object/5 - *s3_session(HTTPTransport)*
- presigned_post_object/4 - *s3_protocol*
- presigned_post_object/5 - *s3_session(HTTPTransport)*
- presigned_put_object/4 - *s3_protocol*
- presigned_put_object/5 - *s3_session(HTTPTransport)*
- presigned_query_pairs/7 - *s3_signing*
- previous/2 - *zipperp*
- previous/3 - *zipperp*
- previous/4 - *dictionaryp*
- previous_arrangement/3 - *arrangements_protocol*
- previous_cartesian_tuple/3 - *cartesian_products_protocol*
- previous_combination/3 - *combinations_protocol*
- previous_derangement/3 - *derangements_protocol*
- previous_distinct_partition/3 - *integer_partitions_protocol*
- previous_distinct_partition/4 - *integer_partitions_protocol*
- previous_multiset/3 - *multisets_protocol*
- previous_partition/3 - *integer_partitions_protocol*, *partitions_protocol*
- previous_partition/4 - *integer_partitions_protocol*, *partitions_protocol*
- previous_permutation/2 - *permutations_protocol*
- primary_office_bic/1 - *iso_9362_protocol*
- print_anomaly_detector/1 - *anomaly_detector_protocol*
- print_association_rule_miner/1 - *association_rule_miner_protocol*
- print_classifier/1 - *classifier_protocol*
- print_clusterer/1 - *clusterer_protocol*
- print_dimension_reducer/1 - *dimension_reducer_protocol*
- print_flags/0 - *flags*, *flags_validator*
- print_flags/1 - *flags*
- print_forecaster/1 - *forecaster_protocol*

- print_message/3 - *logtalk*
- print_message_token/4 - *logtalk*
- print_message_tokens/3 - *logtalk*
- print_pattern_miner/1 - *pattern_miner_protocol*
- print_problem/1 - *linear_programming_protocol*
- print_program/1 - *asp_protocol*, *minizinc_protocol*, *omt_protocol*, *problog_protocol*, *sat_protocol*, *smt_protocol*, *souffle_protocol*
- print_ranker/1 - *ranker_protocol*
- print_recommender/1 - *recommender_protocol*
- print_regressor/1 - *regressor_protocol*
- print_result/1 - *asp_protocol*, *minizinc_protocol*, *omt_protocol*, *problog_protocol*, *sat_protocol*, *smt_protocol*, *souffle_protocol*
- print_results/1 - *omt_protocol*, *smt_protocol*
- print_selector/1 - *feature_selector_protocol*
- print_solution/1 - *linear_programming_protocol*
- probability/3 - *problog_protocol*
- problem_from_matrices/8 - *linear_programming_protocol*
- problem_json_schema/1 - *open_api*
- problem_response/6 - *rest*
- problem_response_descriptor/3 - *open_api*
- problog_statement/2 - *problog_problem_protocol*
- product/2 - *numberlistp*, *statisticsp*
- product/3 - *setp*
- profile/2 - *gravatar*, *language_profiles*
- profile/3 - *gravatar*
- profile_path/2 - *gravatar_api*
- profile_response/3 - *gravatar*
- profile_url/2 - *gravatar*
- program/1 - *asp_protocol*, *minizinc_protocol*, *omt_protocol*, *problog_protocol*, *sat_protocol*, *smt_protocol*, *souffle_protocol*
- program/2 - *asp_protocol*, *minizinc_protocol*, *omt_protocol*, *problog_protocol*, *sat_protocol*, *smt_protocol*, *souffle_protocol*
- program/3 - *sat_protocol*
- program_to_clauses/2 - *metagol*
- progress/5 - *ant_colony_problem_protocol*, *differential_evolution_problem_protocol*, *genetic_algorithm_problem_protocol*, *local_optimization_problem_protocol*, *particle_swarm_optimization_protocol*, *simulated_annealing_protocol*, *tabu_search_problem_protocol*
- prompt/2 - *http_htmx*
- prompt_get/3 - *mcp_prompt_protocol*
- prompt_get_round/4 - *mcp_multiround_protocol*
- prompts/1 - *mcp_prompt_protocol*
- pronouns/2 - *gravatar*
- propagate/2 - *interval_constraint_network_protocol*
- propagate/3 - *interval_constraint_network_protocol*, *tle_orbits_protocol*
- propagate/4 - *tle_orbits_protocol*
- propagate/5 - *tle_orbits_protocol*
- propagate_state/4 - *tle_orbits_protocol*
- propagate_state/5 - *tle_orbits_protocol*
- propagation_triple/2 - *interval_constraint_network_protocol*
- propagation_triples/2 - *interval_constraint_network_protocol*
- proper_prefix/2 - *listp*
- proper_prefix/3 - *listp*

- proper_subsequence/2 - *subsequences_protocol*
- proper_suffix/2 - *listp*
- proper_suffix/3 - *listp*
- properties/2 - *http_websocket_frames*
- property/1 - *application_protocol*, *http_request_protocol*, *http_response_protocol*
- property/2 - *http_core*, *http_sse*, *http_websocket*, *http_websocket_frames*
- property/3 - *open_id_discovery*
- protect_request/4 - *http_authenticate*, *http_digest*, *http_oauth*
- prove/2 - *interpreterp*
- prove/3 - *interpreterp*
- provider/3 - *open_id_discovery*
- provider/4 - *open_id_discovery*
- provides/2 - *registries*
- prune/3 - *c45_classifier*
- prune/5 - *c45_classifier*
- pseudo_inverse/2 - *linear_algebra_protocol*
- pseudo_inverse/3 - *linear_algebra_protocol*
- public_key_pem/2 - *jwt_der*, *jwt_jwk*, *open_id_der*
- public_keypair/2 - *paseto_protocol*
- public_sign/3 - *paseto_protocol*
- public_sign/5 - *paseto_protocol*
- public_verify/3 - *paseto_protocol*
- public_verify/5 - *paseto_protocol*
- publish/4 - *mqtt*
- push_back/3 - *deque_protocol*
- push_front/3 - *deque_protocol*
- put/4 - *generalized_ripple_down_rules*, *http_client*, *lru_cache_protocol*
- put/5 - *http_client*, *http_client_digest_session(HTTPTransport)*, *http_client_session(HTTPTransport)*
- put_object/6 - *s3_protocol*
- put_object/7 - *s3_session(HTTPTransport)*
- qr_decomposition/3 - *linear_algebra_protocol*
- quartiles/4 - *statisticsp*
- query/1 - *datalog_protocol*
- query/2 - *datalog_protocol*
- query/3 - *json_path_protocol*
- query/4 - *http_client*
- query/5 - *http_client*
- query_parameter/3 - *rest*
- question_hook/6 - *logtalk*
- question_prompt_stream/4 - *logtalk*
- queue_bind/4 - *amqp*
- queue_declare/3 - *amqp*
- queue_delete/3 - *amqp*
- queue_purge/2 - *amqp*
- queue_unbind/4 - *amqp*
- quick_check/1 - *lgtunit*
- quick_check/2 - *lgtunit*
- quick_check/3 - *lgtunit*
- quote/3 - *asp_process*, *minizinc_process*, *problog_process*, *sat_process*, *smt_process*, *souffle_process*

- `random/1` - *random_protocol*
- `random/3` - *random_protocol*
- `random_arrangement/3` - *arrangements_protocol*
- `random_below/2` - *crypto*
- `random_bytes/2` - *crypto*
- `random_cartesian_tuple/2` - *cartesian_products_protocol*
- `random_combination/3` - *combinations_protocol*
- `random_derangement/2` - *derangements_protocol*
- `random_distinct_arrangement/3` - *arrangements_protocol*
- `random_distinct_cartesian_tuple/2` - *cartesian_products_protocol*
- `random_distinct_combination/3` - *combinations_protocol*
- `random_distinct_derangement/2` - *derangements_protocol*
- `random_distinct_multiset/3` - *multisets_protocol*
- `random_distinct_partition/2` - *integer_partitions_protocol*, *partitions_protocol*
- `random_distinct_partition/3` - *integer_partitions_protocol*, *partitions_protocol*
- `random_distinct_permutation/2` - *permutations_protocol*
- `random_individual/1` - *genetic_algorithm_problem_protocol*
- `random_multiset/3` - *multisets_protocol*
- `random_node/1` - *uuid_protocol*
- `random_partition/2` - *integer_partitions_protocol*, *partitions_protocol*
- `random_partition/3` - *integer_partitions_protocol*, *partitions_protocol*
- `random_permutation/2` - *permutations_protocol*
- `random_subsequence/2` - *subsequences_protocol*
- `random_tree/1` - *benchmark_generators*
- `randomize/1` - *fast_random(Algorithm)*, *random(Algorithm)*
- `randseq/4` - *random_protocol*
- `randset/4` - *random_protocol*
- `range/2` - *statisticsp*
- `rank/3` - *ranker_protocol*
- `rank_correlation/3` - *statisticsp*
- `ranker_options/2` - *ranker_protocol*
- `rating/3` - *rating_dataset_protocol*
- `rating_count/1` - *rating_dataset_protocol*
- `rating_scale/2` - *rating_dataset_protocol*
- `rd/1` - *linda_client*
- `rd/2` - *linda_client*
- `rd_list/2` - *linda_client*
- `rd_list/3` - *linda_client*
- `rd_noblock/1` - *linda_client*
- `rd_noblock/2` - *linda_client*
- `rdirectories/1` - *lgtdocp*
- `rdirectories/2` - *lgtdocp*
- `rdirectory/1` - *code_metric*, *dead_code_scanner*, *diagram(Format)*, *diagrams(Format)*, *lgtdocp*, *wrapper*
- `rdirectory/2` - *code_metric*, *dead_code_scanner*, *diagram(Format)*, *diagrams(Format)*, *lgtdocp*, *wrapper*
- `rdirectory/3` - *diagram(Format)*, *diagrams(Format)*
- `rdirectory_score/2` - *code_metric*
- `reachable/3` - *graph_protocol*
- `read_file/2` - *csv_protocol*, *read_file*, *tsv_protocol*
- `read_file/3` - *csv_protocol*, *tsv_protocol*

- read_file_by_line/2 - *csv_protocol*, *tsv_protocol*
- read_file_by_line/3 - *csv_protocol*, *tsv_protocol*
- read_frame/2 - *http_websocket_frames*
- read_frame/3 - *http_websocket_frames*
- read_framed_message/2 - *json_rpc*
- read_from_atom/2 - *term_io_protocol*
- read_from_chars/2 - *term_io_protocol*
- read_from_codes/2 - *term_io_protocol*
- read_message/2 - *http_websocket_messages(TextRepresentation)*, *json_rpc*
- read_message/4 - *http_websocket_session(Role,TextRepresentation)*
- read_message/5 - *http_websocket_session(Role,TextRepresentation)*
- read_message/6 - *http_websocket_session(Role,TextRepresentation)*
- read_mime_types/2 - *mime_types*
- read_only_device_path/1 - *osp*
- read_request/2 - *http_server_core*
- read_response/2 - *http_client_core*
- read_response_head/2 - *http_client_core*
- read_stream/2 - *csv_protocol*, *tsv_protocol*
- read_stream/3 - *csv_protocol*, *tsv_protocol*
- read_stream_by_line/2 - *csv_protocol*, *tsv_protocol*
- read_stream_by_line/3 - *csv_protocol*, *tsv_protocol*
- read_term_from_atom/3 - *term_io_protocol*
- read_term_from_chars/3 - *term_io_protocol*
- read_term_from_chars/4 - *term_io_protocol*
- read_term_from_codes/3 - *term_io_protocol*
- read_term_from_codes/4 - *term_io_protocol*
- readme/1 - *packs_common*
- readme/2 - *packs_common*
- reason_code/3 - *mqtt*
- reassemble_aos_frames/5 - *ccsds_packet_services*
- reassemble_aos_frames/6 - *ccsds_packet_services*
- reassemble_aos_frames/7 - *ccsds_packet_services*
- reassemble_aos_packets/5 - *ccsds_packet_services*
- reassemble_aos_packets/6 - *ccsds_packet_services*
- reassemble_aos_packets/7 - *ccsds_packet_services*
- reassemble_frames/6 - *ccsds_link_profiles*
- reassemble_frames/7 - *ccsds_link_profiles*
- reassemble_frames/8 - *ccsds_link_profiles*
- reassemble_packet_zone/5 - *ccsds_packet_services*
- reassemble_packets/6 - *ccsds_link_profiles*
- reassemble_packets/7 - *ccsds_link_profiles*
- reassemble_packets/8 - *ccsds_link_profiles*
- reassemble_tc_frame/4 - *ccsds_tc_services*
- reassemble_tc_frame/5 - *ccsds_tc_services*
- reassemble_tc_frame/6 - *ccsds_tc_services*
- reassemble_tc_frame_with_provenance/5 - *ccsds_tc_services*
- reassemble_tc_frame_with_provenance/6 - *ccsds_tc_services*
- reassemble_tc_frames/4 - *ccsds_tc_services*
- reassemble_tc_frames/5 - *ccsds_tc_services*

- `reassemble_tc_frames/6` - *ccsds_tc_services*
- `reassemble_tc_frames_with_provenance/5` - *ccsds_tc_services*
- `reassemble_tc_frames_with_provenance/6` - *ccsds_tc_services*
- `reassemble_tm_frames/5` - *ccsds_packet_services*
- `reassemble_tm_frames/6` - *ccsds_packet_services*
- `reassemble_tm_frames/7` - *ccsds_packet_services*
- `reassemble_tm_packets/5` - *ccsds_packet_services*
- `reassemble_tm_packets/6` - *ccsds_packet_services*
- `reassemble_tm_packets/7` - *ccsds_packet_services*
- `receive/2` - *http_sse*, *http_websocket*
- `receive/3` - *amqp*, *http_sse*, *http_websocket*, *mqtt*, *stomp*
- `receive_binary/2` - *http_websocket*
- `receive_data/2` - *http_sse*
- `receive_json/2` - *http_sse*, *http_websocket*
- `receive_term/2` - *http_sse*, *http_websocket*
- `receive_text/2` - *http_websocket*
- `recommend/4` - *recommender_protocol*
- `recommender_options/2` - *recommender_protocol*
- `recorda/2` - *recorded_database_core*
- `recorda/3` - *recorded_database_core*
- `recorded/2` - *recorded_database_core*
- `recorded/3` - *recorded_database_core*
- `recordz/2` - *recorded_database_core*
- `recordz/3` - *recorded_database_core*
- `reference_kind/2` - *url(Representation)*
- `refine/5` - *interval_constraint_network_protocol*
- `refine_propagate/3` - *interval_constraint_network_protocol*
- `refine_propagate/4` - *interval_constraint_network_protocol*
- `refine_propagate/5` - *interval_constraint_network_protocol*
- `refine_propagate/6` - *interval_constraint_network_protocol*
- `refresh_token/4` - *open_id*, *open_id_client*
- `register/2` - *http_websocket_service_registry*
- `regressor_options/2` - *regressor_protocol*
- `relation/1` - *interval_algebra_protocol*
- `relation/3` - *interval_protocol*
- `relation/4` - *interval_constraint_network_protocol*, *souffle_protocol*
- `relation_set/1` - *interval_relation_set_protocol*
- `relative_standard_deviation/2` - *statisticsp*
- `relativize/3` - *url(Representation)*
- `release/1` - *amqp_pool*
- `release_date/1` - *application_protocol*
- `relevance/3` - *ranking_dataset_protocol*
- `remote/3` - *git_protocol*
- `remove/3` - *generalized_ripple_down_rules*, *http_server_core_session*
- `removeDependent/1` - *subject*
- `remove_catalog/3` - *bm25_recommender*
- `remove_constraints/3` - *stn_protocol*
- `remove_diacritics/2` - *text_normalizer_protocol*
- `remove_duplicates/2` - *listp*, *varlistp*

- remove_ratings/3 - *bm25_recommender*, *jaccard_recommender*, *tfidf_recommender*
- remove_rule/1 - *datalog_protocol*
- remove_time_points/3 - *stn_protocol*
- rename/4 - *redis*
- rename_file/2 - *osp*
- renew/2 - *http_server_core_session*
- replace/3 - *zipperp*
- replace/5 - *memcached*
- replace_content/3 - *bm25_recommender*, *jaccard_recommender*, *tfidf_recommender*
- replace_sub_atom/4 - *atom*
- reply/3 - *http_htmx*
- reply/4 - *http_htmx*
- report_directory/3 - *mutation_testing*
- report_entity/3 - *mutation_testing*
- report_library/3 - *mutation_testing*
- report_predicate/4 - *mutation_testing*
- repository/1 - *application_protocol*
- repository_branch/1 - *application_protocol*
- repository_commit/1 - *application_protocol*
- repository_commit_abbreviated/1 - *application_protocol*
- repository_commit_author/1 - *application_protocol*
- repository_commit_date/1 - *application_protocol*
- repository_commit_message/1 - *application_protocol*
- representation/2 - *atms_protocol*
- request/3 - *json_rpc*
- request/4 - *http_client*, *json_rpc*
- request/5 - *http_client*, *http_client_digest_session(HTTPTransport)*, *http_client_session(HTTPTransport)*, *open_ai_client*
- request/6 - *s3_endpoints*
- request/7 - *http_core*
- request_body/2 - *rest*
- request_body_bytes/4 - *http_core*
- request_cookies/3 - *http_cookie_jar*
- request_cookies/4 - *http_cookie_jar*
- request_header/3 - *rest*
- request_kind/2 - *http_htmx*
- request_listener_shutdown/1 - *http_server*, *http_transport_protocol*
- request_origin/2 - *http_cors*
- request_properties/2 - *http_htmx*
- request_property/2 - *http_htmx*
- request_shutdown/1 - *http_server*, *http_transport_protocol*, *http_websocket_server_service(HTTPTransport)*
- rescale/3 - *numberlistp*
- reset/0 - *counter*, *debuggerp*, *linter_reporter*, *mime_types*, *mutator_protocol*, *packs_common*, *ports_profiler*
- reset/1 - *ports_profiler*
- reset_counter/1 - *counters*
- reset_counters/0 - *counters*
- reset_flags/0 - *flags*
- reset_flags/1 - *flags*
- reset_genint/0 - *genint_core*
- reset_genint/1 - *genint_core*

- `reset_gensym/0` - *gensym_core*
- `reset_gensym/1` - *gensym_core*
- `reset_monitor/0` - *monitorp*
- `reset_seed/0` - *fast_random(Algorithm)*, *random(Algorithm)*
- `residuals/2` - *hodge_rank*
- `resize/2` - *amqp_pool*
- `resolve/3` - *url(Representation)*
- `resolve_command_path/2` - *osp*
- `resource_read/3` - *mcp_resource_protocol*
- `resource_read_round/4` - *mcp_multiround_protocol*
- `resource_templates/1` - *mcp_resource_protocol*
- `resource_ui_meta/2` - *mcp_ui_protocol*
- `resources/1` - *mcp_resource_protocol*
- `response/3` - *http_oauth_metadata*, *json_rpc*
- `response/4` - *http_oauth_metadata*
- `response/6` - *http_core*
- `response_body_bytes/4` - *http_core*
- `response_producing/1` - *smt_program*
- `response_properties/2` - *s3_headers*
- `restart_state/2` - *tabu_search_problem_protocol*
- `restore/1` - *packs*
- `restore/2` - *packs*
- `result/2` - *json_rpc*
- `result_property/2` - *asp_protocol*, *minizinc_protocol*, *omt_protocol*, *problog_protocol*, *sat_protocol*, *smt_protocol*, *souffle_protocol*
- `retract_batch/4` - *atms_protocol*
- `retract_fact/1` - *datalog_protocol*
- `retract_justification/5` - *atms_protocol*
- `retract_justifications/3` - *atms_protocol*
- `retract_node/3` - *atms_protocol*
- `retract_nodes/3` - *atms_protocol*
- `reverse/2` - *listp*, *varlistp*
- `revise/6` - *ripple_down_rules_common_protocol*
- `rewind/2` - *zipperp*
- `rewind/3` - *zipperp*
- `rhumb_bearing/3` - *geospatial_protocol*
- `rhumb_destination_point/4` - *geospatial_protocol*
- `rhumb_distance/3` - *geospatial_protocol*
- `rhumb_midpoint/3` - *geospatial_protocol*
- `rising_factorial/3` - *natural*
- `rlibraries/1` - *lgtdocp*
- `rlibraries/2` - *lgtdocp*
- `rlibrary/1` - *code_metric*, *dead_code_scanner*, *diagram(Format)*, *diagrams(Format)*, *lgtdocp*
- `rlibrary/2` - *code_metric*, *dead_code_scanner*, *diagram(Format)*, *diagrams(Format)*, *lgtdocp*
- `rlibrary_score/2` - *code_metric*
- `rol32/3` - *hash_common_32*
- `rol64/3` - *hash_common_64*
- `rollback/0` - *datalog_protocol*
- `root/2` - *git_protocol*

- root_mean_squared_error/3 - *statisticsp*
- ror32/3 - *hash_common_32*
- route_distance/2 - *geospatial_protocol*
- route_distance/3 - *geospatial_protocol*
- route_distance/4 - *geospatial_protocol*
- route_parameters/2 - *http_router_parameters*
- rpop/3 - *redis*
- rpush/4 - *redis*
- rule/2 - *databasep*
- rule/3 - *databasep*
- rule/4 - *databasep*
- rule_metric/3 - *association_rule_miner_protocol*
- rules/1 - *datalog_protocol*
- run/0 - *lgtunit*
- run/1 - *lgtunit*
- run/2 - *ant_colony(Problem,Random)*, *differential_evolution(Problem,Random)*, *genetic_algorithm(Problem,Random)*, *lgtunit*, *local_optimization_solver(Problem)*, *particle_swarm_optimization(Problem,Random)*, *simulated_annealing(Problem,Random)*, *tabu_search(Problem,Random)*
- run/3 - *ant_colony(Problem,Random)*, *differential_evolution(Problem,Random)*, *genetic_algorithm(Problem,Random)*, *local_optimization_solver(Problem)*, *particle_swarm_optimization(Problem,Random)*, *simulated_annealing(Problem,Random)*, *tabu_search(Problem,Random)*
- run/4 - *ant_colony(Problem,Random)*, *differential_evolution(Problem,Random)*, *genetic_algorithm(Problem,Random)*, *local_optimization_solver(Problem)*, *particle_swarm_optimization(Problem,Random)*, *simulated_annealing(Problem,Random)*, *tabu_search(Problem,Random)*
- run/7 - *asp_process*, *problog_process*, *sat_process*, *smt_process*
- run/8 - *minizinc_process*
- run/10 - *souffle_process*
- run_session/3 - *http_websocket_service(HTTPTransport,Role,TextRepresentation)*
- run_session/4 - *http_websocket_service(HTTPTransport,Role,TextRepresentation)*
- run_stdio_loop/3 - *mcp_server_2026_07_28_spec*
- run_test_sets/1 - *lgtunit*
- sadd/4 - *redis*
- same_instant/2 - *datep*
- same_length/2 - *listp*, *varlistp*
- same_length/3 - *listp*
- sample/3 - *random_protocol*
- sample/4 - *random_protocol*
- sample_arrangements/4 - *arrangements_protocol*
- sample_cartesian_tuples/3 - *cartesian_products_protocol*
- sample_combinations/4 - *combinations_protocol*
- sample_derangements/3 - *derangements_protocol*
- sample_distinct_arrangements/4 - *arrangements_protocol*
- sample_distinct_cartesian_tuples/3 - *cartesian_products_protocol*
- sample_distinct_combinations/4 - *combinations_protocol*
- sample_distinct_derangements/3 - *derangements_protocol*
- sample_distinct_multisets/4 - *multisets_protocol*
- sample_distinct_partitions/3 - *integer_partitions_protocol*, *partitions_protocol*
- sample_distinct_partitions/4 - *integer_partitions_protocol*, *partitions_protocol*
- sample_distinct_permutations/3 - *permutations_protocol*
- sample_multisets/4 - *multisets_protocol*

- sample_partitions/3 - *integer_partitions_protocol*, *partitions_protocol*
- sample_partitions/4 - *integer_partitions_protocol*, *partitions_protocol*
- sample_permutations/3 - *permutations_protocol*
- save/0 - *wrapper*
- save/1 - *packs*, *tzif_protocol*, *wrapper*
- save/2 - *http_cookie_jar*, *packs*, *tzif_protocol*
- save_dut1_entries/1 - *time_scales_data*, *time_scales_protocol*
- save_leap_seconds_entries/1 - *time_scales_data*, *time_scales_protocol*
- scalar_product/3 - *numberlistp*
- scale_matrix/3 - *linear_algebra_protocol*
- scale_vector/3 - *linear_algebra_protocol*
- scan_left/4 - *metap*
- scan_left_1/3 - *metap*
- scan_right/4 - *metap*
- scan_right_1/3 - *metap*
- scard/3 - *redis*
- schedule/2 - *stn_protocol*
- schema/2 - *http_router*, *open_api_provider_protocol*
- scopes/2 - *http_oauth*
- score/3 - *anomaly_detector_protocol*, *feature_scoring_protocol*, *isolation_forest_anomaly_detector*
- score/4 - *recommender_protocol*
- score_all/3 - *anomaly_detector_protocol*, *isolation_forest_anomaly_detector*
- score_all/4 - *bm25_recommender*, *jaccard_recommender*, *tfidf_recommender*
- score_content/4 - *bm25_recommender*, *jaccard_recommender*, *tfidf_recommender*
- scores/2 - *ranker_protocol*
- scores/3 - *language_detection_strategy_protocol*
- script_ratios/2 - *language_detection_scripts*
- scripts_to_languages/2 - *language_detection_scripts*
- search/1 - *packs*
- secondary_header/2 - *ccsds_packets(SecondaryHeaderLength)*, *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- secondary_header_flag/2 - *ccsds_packets(SecondaryHeaderLength)*, *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,Ha*
- secondary_header_time/3 - *ccsds_packets(SecondaryHeaderLength)*
- section_visibility/2 - *gravatar*
- secure_compare/2 - *crypto*
- security/1 - *http_router*, *open_api_provider_protocol*
- security_scheme/2 - *http_router*, *open_api_provider_protocol*
- segment_data/2 - *ccsds_tc_services*
- segment_header/2 - *ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF)*
- segment_header_suffix/2 - *ccsds_tc_services*
- segment_header_suffixes/2 - *ccsds_tc_services*
- segment_length_identifier/2 - *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- segments/5 - *jwt_compact*
- select/3 - *listp*, *random_protocol*, *setp*, *varlistp*
- select/4 - *listp*, *random_protocol*
- select_key/3 - *jwt_jwks*, *open_id_jwks*
- select_key/4 - *jwt_jwks*
- select_keys/4 - *jwt_jwks*, *paseto_keys*
- selectchk/3 - *listp*, *setp*
- selectchk/4 - *listp*

- selected_features/2 - *feature_selector_protocol*
- selector_options/2 - *feature_selector_protocol*
- send/2 - *http_sse*, *http_websocket*
- send/3 - *http_sse*, *http_websocket*, *http_websocket_service_registry*, *redis*
- send/4 - *smtp_protocol*, *stomp*
- send/5 - *smtp_protocol*
- send_binary/2 - *http_websocket*
- send_comment/2 - *http_sse*
- send_data/2 - *http_sse*
- send_event/3 - *http_sse*
- send_heartbeat/1 - *amqp*, *stomp*
- send_id_data/3 - *http_sse*
- send_json/2 - *http_sse*, *http_websocket*
- send_pingreq/1 - *mqtt*
- send_retry/2 - *http_sse*
- send_term/2 - *http_sse*, *http_websocket*
- send_text/2 - *http_websocket*
- sentence_type/2 - *nmea_protocol*
- sequence/2 - *either*, *maybe*, *sequence_dataset_protocol*, *validated*
- sequence/3 - *integer*
- sequence/4 - *float*, *integer*, *random_protocol*
- sequence/5 - *float*
- sequence_count/2 - *ccsds_packets(SecondaryHeaderLength)*
- sequence_flags/2 - *ccsds_packets(SecondaryHeaderLength)*, *ccsds_tc_services*
- sequence_number/2 - *ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF)*
- sequential_occurrences/2 - *listp*
- sequential_occurrences/3 - *listp*
- serialize/2 - *asp_program*, *minizinc_program*, *problog_program*, *sat_program*, *smt_program*, *souffle_program*
- serialize/4 - *asp_program*, *minizinc_program*, *problog_program*, *sat_program*, *smt_program*, *souffle_program*
- series_length/1 - *time_series_dataset_protocol*
- serve/3 - *http_server_core*, *queuep*
- serve/4 - *http_directory_listing*, *http_server*, *http_static_files*
- serve/5 - *http_directory_listing*, *http_server*, *http_static_files*
- serve/6 - *http_server*
- serve_connection/3 - *http_server_core*
- serve_listener/4 - *http_transport_protocol*
- serve_listener/5 - *http_transport_protocol*
- serve_once/3 - *http_server*, *http_transport_protocol*
- serve_once/5 - *http_sse*, *http_websocket*
- serve_once/6 - *http_sse*, *http_websocket*, *http_websocket_server_service(HTTPTransport)*
- serve_once/7 - *http_websocket_server_service(HTTPTransport)*
- serve_sse/4 - *http_server_core*
- serve_sse_once/5 - *http_transport_protocol*
- serve_until_shutdown/4 - *http_transport_protocol*
- serve_until_shutdown/5 - *http_server*, *http_transport_protocol*, *http_websocket_server_service(HTTPTransport)*
- serve_until_shutdown/6 - *http_server*, *http_websocket_server_service(HTTPTransport)*
- serve_websocket/4 - *http_server_core*
- serve_websocket_once/5 - *http_server*, *http_transport_protocol*
- server_accept/4 - *socket*

- server_accept/5 - *socket*
- server_close/1 - *socket*
- server_open/2 - *socket*
- server_open/3 - *socket*
- server_open/4 - *socket*
- server_property/2 - *http_server*
- servers/1 - *http_router*, *open_api_provider_protocol*
- session/2 - *http_websocket_service_registry*
- session_count/2 - *http_websocket_service_registry*
- sessions/2 - *http_websocket_service_registry*
- set/1 - *counter*
- set/3 - *http_server_core_session*, *memcached*
- set/4 - *random_protocol*, *redis*
- set/5 - *memcached*
- set_element/2 - *java_utils_protocol*
- set_field/2 - *java_access_protocol*
- set_flag_value/2 - *flags*
- set_flag_value/3 - *flags*
- set_monitor/4 - *event_registryp*
- set_rating_scale/3 - *bm25_recommender*
- set_seed/1 - *arbitrary*, *pseudo_random_protocol*
- set_spy_point/4 - *monitorp*
- set_write_max_depth/1 - *debuggerp*
- setrange/5 - *redis*
- setup/0 - *packs_common*
- shell/1 - *osp*
- shell/2 - *osp*
- shell_command/1 - *doclet*
- shift_matrix_diagonal/3 - *linear_algebra_protocol*
- shl64/3 - *hash_common_64*
- short_flags/1 - *command_line_option*
- shr64/3 - *hash_common_64*
- shrink/3 - *arbitrary*
- shrink_sequence/3 - *arbitrary*
- shrinker/1 - *arbitrary*
- shutdown_server/1 - *linda_client*
- sign/4 - *paseto*
- sign/5 - *jwt*, *jwt_jws*
- sign//1 - *number_grammars(Format)*
- sign_payload/5 - *jwt*, *jwt_jws*
- sign_request/8 - *s3_signing*
- signaling_field/2 - *ccsds_aos_frames(FrameLength,InsertZoneLength,HasOCF,HasFECF)*
- signed_integer_to_bytes/4 - *byte_order*
- signed_integer_to_bytes/5 - *byte_order*
- signing_input/4 - *jwt_compact*
- similarity/3 - *similarity_metric_protocol*
- singleton/2 - *atms_environment_protocol*, *interval_relation_set_protocol*
- singletons/2 - *termp*
- sink_closest_min_cut/5 - *max_flow_protocol*

- sismember/4 - *redis*
- size/2 - *dictionaryp*, *heap_protocol*, *lru_cache_protocol*, *ripple_down_rules_common_protocol*, *setp*, *trie(Representation)*
- size/3 - *souffle_protocol*
- skewness/2 - *statisticsp*
- sleep/1 - *osp*
- sliding_window/3 - *subsequences_protocol*
- smembers/3 - *redis*
- smt_statement/2 - *smt_problem_protocol*
- softmax/2 - *numberlistp*
- softmax/3 - *numberlistp*
- software_heritage_identifier/1 - *application_protocol*
- solution_cost/2 - *ant_colony_problem_protocol*
- solve/2 - *asp_protocol*, *linear_programming_protocol*, *minizinc_protocol*, *omt_protocol*, *problog_protocol*, *sat_protocol*, *smt_protocol*, *souffle_protocol*
- solve/3 - *asp_protocol*, *linear_programming_protocol*, *minizinc_protocol*, *omt_protocol*, *problog_protocol*, *sat_protocol*, *smt_protocol*, *souffle_protocol*
- solve/4 - *ode_solver_protocol*
- solve/5 - *ode_solver_protocol*
- solve/6 - *ode_solver_protocol*
- solve/8 - *qp_solver_protocol*
- solve_cholesky/3 - *linear_algebra_protocol*
- solve_linear_system/3 - *linear_algebra_protocol*
- solve_linear_systems/3 - *linear_algebra_protocol*
- solve_lower_triangular_matrix/3 - *linear_algebra_protocol*
- solve_upper_triangular/3 - *linear_algebra_protocol*
- solve_upper_triangular_matrix/3 - *linear_algebra_protocol*
- sort/2 - *listp*
- sort/3 - *listp*
- sort/4 - *listp*
- souffle_statement/2 - *souffle_problem_protocol*
- soundex/2 - *string_distance(Representation)*
- soundex_match/2 - *string_distance(Representation)*
- source_file_extension/1 - *modules_diagram_support*
- space//0 - *blank_grammars(Format)*
- spacecraft_id/2 - *ccsds_frame_protocol*, *ccsds_frames*
- spaces//0 - *blank_grammars(Format)*
- spec/1 - *mcp_server_spec_protocol*, *mcp_server_transport_protocol*
- spec_version/1 - *open_ai_catalog*
- split/3 - *atom*
- split/4 - *listp*
- split_aos_packet_zone/3 - *ccsds_packet_services*
- split_packet_zone/4 - *ccsds_packet_services*
- split_sentence_codes/3 - *tokenizer_language_protocol*
- split_sentences/2 - *sentence_splitter_protocol*
- split_sentences/3 - *sentence_splitter_protocol*
- split_string/4 - *string(Representation)*
- spy/1 - *debuggerp*
- spy/3 - *debuggerp*
- spy/4 - *debuggerp*

- `spy_point/4` - *monitorp*
- `spying/1` - *debuggerp*
- `spying/3` - *debuggerp*
- `spying/4` - *debuggerp*
- `squared_mahalanobis_distance/4` - *multivariate_distributions_protocol*
- `squared_mahalanobis_distance/5` - *multivariate_distributions_protocol*
- `srem/4` - *redis*
- `sse_headers/1` - *mcp_server_streamable_http_transport*
- `stabilize_vector_sign/2` - *linear_algebra_protocol*
- `stabilize_vector_sign/3` - *linear_algebra_protocol*
- `standard_cauchy/3` - *sampling_protocol*
- `standard_deviation/2` - *statisticsp*
- `standard_error/2` - *statisticsp*
- `standard_exponential/1` - *sampling_protocol*
- `standard_gamma/2` - *sampling_protocol*
- `standard_normal/1` - *sampling_protocol*, *univariate_distributions_protocol*
- `standard_normal_density/2` - *univariate_distributions_protocol*
- `standard_normal_distribution/2` - *univariate_distributions_protocol*
- `standard_normal_log_density/2` - *univariate_distributions_protocol*
- `standard_normal_quantile/2` - *univariate_distributions_protocol*
- `standard_normal_samples/2` - *univariate_distributions_protocol*
- `standard_t/2` - *sampling_protocol*, *univariate_distributions_protocol*
- `standard_t_density/3` - *univariate_distributions_protocol*
- `standard_t_distribution/3` - *univariate_distributions_protocol*
- `standard_t_log_density/3` - *univariate_distributions_protocol*
- `standard_t_quantile/3` - *univariate_distributions_protocol*
- `standard_t_samples/3` - *univariate_distributions_protocol*
- `start/0` - *ports_profiler*, *shell*
- `start/2` - *mcp_server*
- `start/3` - *mcp_server*
- `start/4` - *http_server*, *mcp_server*, *mcp_server_transport_protocol*
- `start/5` - *http_server*, *mcp_server*
- `start_redirect_to_file/2` - *dump_trace*
- `started_by/2` - *interval_protocol*
- `starts/2` - *interval_protocol*
- `state_energy/2` - *genetic_algorithm_problem_protocol*, *simulated_annealing_protocol*, *tabu_search_problem_protocol*
- `statistics/2` - *linear_programming_protocol*
- `stats/1` - *amqp_pool*
- `stats/2` - *memcached*
- `stats/3` - *memcached*
- `status/1` - *http_response_protocol*
- `status/2` - *http_core*, *linear_programming_protocol*, *omt_protocol*, *smt_protocol*
- `stem/2` - *stemmer_protocol*
- `stems/2` - *stemmer_protocol*
- `stirling_first/3` - *natural*
- `stirling_second/3` - *natural*
- `stop/0` - *ports_profiler*
- `stop/1` - *http_server*

- stop_condition/3 - *ant_colony_problem_protocol*, *differential_evolution_problem_protocol*, *genetic_algorithm_problem_protocol*, *local_optimization_problem_protocol*, *particle_swarm_optimization_protocol*, *simulated_annealing_protocol*, *tabu_search_problem_protocol*
- stop_redirect_to_file/0 - *dump_trace*
- stop_word/1 - *language_profile_protocol*, *stop_words_language_protocol*, *stop_words_protocol*
- store_set_cookies/3 - *http_cookie_jar*
- strata/1 - *datalog_protocol*
- stream_to_bytes/2 - *reader*
- stream_to_bytes/3 - *reader*
- stream_to_chars/2 - *reader*
- stream_to_chars/3 - *reader*
- stream_to_codes/2 - *reader*
- stream_to_codes/3 - *reader*
- stream_to_terms/2 - *reader*
- stream_to_terms/3 - *reader*
- streams/3 - *smtp_process_transport*
- string_chars/2 - *string(Representation)*
- string_codes/2 - *string(Representation)*
- string_concat/3 - *string(Representation)*
- string_length/2 - *string(Representation)*
- string_lower/2 - *string(Representation)*
- string_upper/2 - *string(Representation)*
- strings/2 - *trie(Representation)*
- strings/3 - *trie(Representation)*
- strlen/3 - *redis*
- strongest_paths/2 - *schulze_ranker*
- strongly_connected_components/2 - *directed_graph_protocol*
- sub_string/5 - *string(Representation)*
- subclass/1 - *class_hierarchyp*
- subclasses/1 - *class_hierarchyp*
- subdivision/4 - *iso_3166_2*, *iso_3166_protocol*
- subfactorial/2 - *natural*
- sublist/2 - *listp*, *varlistp*
- submilliseconds/2 - *ccsds_cds(DaySegmentOctets,SubmillisecondOctets,Epoch)*
- subscribe/4 - *mqtt*, *stomp*
- subsequence/2 - *subsequences_protocol*
- subsequence/3 - *listp*, *subsequences_protocol*
- subsequence/4 - *listp*
- subsequence_at_indices/3 - *subsequences_protocol*
- subsequence_length/2 - *subsequences_protocol*
- subsequences/2 - *subsequences_protocol*
- subsequences/3 - *subsequences_protocol*
- subsequences_with_min_span/3 - *subsequences_protocol*
- subset/2 - *atms_environment_protocol*, *interval_relation_set_protocol*, *setp*
- subslices/2 - *subsequences_protocol*
- substitute/4 - *listp*
- subsumes/2 - *interval_constraint_network_protocol*, *termp*
- subterm/2 - *termp*, *xml*
- subtract/3 - *listp*, *setp*, *varlistp*

- subtract_duration/3 - *datep*
- subtract_matrices/3 - *linear_algebra_protocol*
- subtract_vectors/3 - *linear_algebra_protocol*
- succ/2 - *integer*
- sufficient/2 - *http_oauth_scope_checker_protocol*
- suffix/2 - *listp*, *typeid_protocol*, *varlistp*
- suffix/3 - *listp*
- suffix_alias/2 - *mime_types*
- suffix_data/2 - *ccsds_packet_services*
- sum/2 - *numberlistp*, *statisticsp*
- sum_of_squares/2 - *statisticsp*
- summary/1 - *linter_reporter*
- superclass/1 - *class_hierarchyp*
- superclasses/1 - *class_hierarchyp*
- supplier/1 - *application_protocol*
- supported_algorithm/1 - *jwt_jwa*
- supported_range/2 - *time_scales_protocol*
- supported_request_scheme/1 - *http_transport_protocol*
- supported_specs/1 - *mcp_server_2025_06_18_spec*
- supported_websocket_scheme/1 - *http_transport_protocol*
- suspend_monitor/0 - *monitorp*
- swap/1 - *expected(Expected)*, *validation(Validation)*
- swap/2 - *random_protocol*
- swap_consecutive/2 - *random_protocol*
- symbol_codes/2 - *asp_program*
- symdiff/3 - *setp*
- symmetric_closure/2 - *directed_graph_protocol*
- symmetric_eigen/3 - *linear_algebra_protocol*
- symmetric_eigen/4 - *linear_algebra_protocol*
- symmetric_eigen/5 - *linear_algebra_protocol*
- symmetric_eigen_partial/7 - *linear_algebra_protocol*
- symmetric_key_bytes/2 - *jwt_jwk*
- synchronization_flag/2 - *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- t/4 - *univariate_distributions_protocol*
- t_density/5 - *univariate_distributions_protocol*
- t_distribution/5 - *univariate_distributions_protocol*
- t_log_density/5 - *univariate_distributions_protocol*
- t_quantile/5 - *univariate_distributions_protocol*
- t_samples/5 - *univariate_distributions_protocol*
- tab//0 - *blank_grammars(Format)*
- tabs//0 - *blank_grammars(Format)*
- tabu_key/2 - *tabu_search_problem_protocol*
- tabu_tenure/4 - *tabu_search_problem_protocol*
- tai_minus_utc_for_tai_unix/2 - *time_scales_data*
- tail/2 - *subsequences_protocol*
- tail1/2 - *subsequences_protocol*
- tails/2 - *subsequences_protocol*
- tails1/2 - *subsequences_protocol*
- take/3 - *listp*

- take/4 - *listp*
- take_pending/3 - *http_websocket_service_registry*
- talker/2 - *nmea_protocol*
- target/1 - *http_request_protocol*, *regression_dataset_protocol*, *target_supervised_dimension_reduction_dataset_protocol*
- target/2 - *http_core*, *http_htmx*
- tcb_minus_tdb_approx/3 - *time_scales_data*
- tcg_minus_tt_approx/3 - *time_scales_data*
- tdb_minus_tt_approx/3 - *time_scales_data*
- team/3 - *multiplayer_ranking_dataset_protocol*
- team_member/4 - *multiplayer_ranking_dataset_protocol*
- temporal_pairwise_dataset_connected_components/2 - *ranking_dataset_common*
- temporal_pairwise_dataset_declared_items/2 - *ranking_dataset_common*
- temporal_pairwise_dataset_games/2 - *ranking_dataset_common*
- temporal_pairwise_dataset_games/3 - *ranking_dataset_common*
- temporal_pairwise_dataset_items/2 - *ranking_dataset_common*
- temporal_pairwise_dataset_periods/2 - *ranking_dataset_common*
- temporal_pairwise_dataset_summary/2 - *ranking_dataset_common*
- temporary_directory/1 - *osp*
- temporary_tls_credentials/3 - *http_process_transport*
- temporary_tls_credentials_files/3 - *http_process_transport*
- term/2 - *sarif*
- term/4 - *sarif*
- term_expansion/2 - *expanding*
- terms_to_array/2 - *java_utils_protocol*
- test/1 - *lgtunit*
- text_body/2 - *rest*
- time_of_day/4 - *ccsds_ccs(CalendarVariant,FractionOctets)*
- time_points/2 - *stn_protocol*
- time_stamp/1 - *osp*
- time_string/3 - *iso8601*
- time_type/2 - *tzif_protocol*
- time_type/3 - *tzif_protocol*
- time_type/4 - *tzif_protocol*
- timeout/1 - *metagol*
- timestamp/2 - *ulid_protocol*
- timestamp/8 - *ulid_protocol*
- title_case/2 - *text_normalizer_protocol*
- to_expected/1 - *validation(Validation)*
- to_expected/2 - *optional(Optional)*
- to_list/2 - *atms_environment_protocol*
- to_optional/1 - *expected(Expected)*, *validation(Validation)*
- to_uuid/2 - *typeid_protocol*
- today/3 - *datep*
- token_hex/2 - *crypto*
- token_info/2 - *http_oauth*
- token_urlsafe/2 - *crypto*
- tokenize/2 - *tokenizer_protocol*
- tokenize/3 - *tokenizer_protocol*
- tokenize_codes/3 - *tokenizer_language_protocol*

- `tokenize_sentences/2` - *tokenizer_protocol*
- `tokenize_sentences/3` - *tokenizer_protocol*
- `tolerance_equal/4` - *lgtunit*, *number*
- `tool/1` - *help*
- `tool_call/3` - *mcp_tool_protocol*
- `tool_call/4` - *mcp_tool_protocol*
- `tool_call_round/4` - *mcp_multiround_protocol*
- `tool_descriptors_to_json/3` - *mcp_server_application*
- `tool_ui/2` - *mcp_ui_protocol*
- `tools/0` - *help*
- `tools/1` - *mcp_tool_protocol*
- `top/3` - *heap_protocol*
- `top_next/5` - *heap_protocol*
- `topological_sort/2` - *directed_graph_protocol*
- `topological_sort/3` - *unweighted_directed_graph(Dictionary)*
- `totp/5` - *otp_protocol*
- `totp_verify/7` - *otp_protocol*
- `touch/3` - *memcached*
- `trace/0` - *debuggerp*
- `trace_event/2` - *logtalk*
- `transaction/2` - *transaction_dataset_protocol*
- `transform/3` - *dimension_reducer_protocol*, *text_vectorizer_protocol*
- `transform/4` - *crs_projections_protocol*
- `transform_all/3` - *text_vectorizer_protocol*
- `transitive_closure/2` - *directed_graph_protocol*
- `transitive_reduction/2` - *unweighted_directed_graph(Dictionary)*
- `transpose/2` - *directed_graph_protocol*, *pairs*
- `transpose_matrix/2` - *linear_algebra_protocol*
- `traverse/3` - *either*, *maybe*, *validated*
- `tree/2` - *gomory_hu(Dictionary)*
- `triangular/4` - *sampling_protocol*
- `trigger/2` - *http_htmx*
- `trigger_name/2` - *http_htmx*
- `trigram_counts/1` - *language_profile_common*, *language_profile_protocol*
- `trigram_counts/2` - *language_profile_common*
- `trigrams/2` - *n_grams_protocol*
- `trim/2` - *string(Representation)*
- `trim/3` - *string(Representation)*
- `trim_left/2` - *string(Representation)*
- `trim_left/3` - *string(Representation)*
- `trim_right/2` - *string(Representation)*
- `trim_right/3` - *string(Representation)*
- `trimmed_mean/3` - *statisticsp*
- `true/1` - *java_utils_protocol*
- `try_add_constraints/4` - *stn_protocol*
- `tt_minus_tai/2` - *time_scales_data*
- `ttl/3` - *redis*
- `tx_commit/1` - *amqp*
- `tx_rollback/1` - *amqp*

- `tx_select/1` - *amqp*
- `type/1` - *command_line_option*, *type*
- `type/2` - *ccsds_packets(SecondaryHeaderLength)*, *http_websocket_messages(TextRepresentation)*
- `type/3` - *redis*
- `tzdb_version/1` - *tzif_zone_ids*
- `unauthorized_response/3` - *http_authenticate*, *http_digest*, *http_oauth*
- `unauthorized_response/4` - *http_authenticate*, *http_digest*, *http_oauth*
- `unexpected/1` - *expected(Expected)*
- `unexpecteds/2` - *either*
- `uniform/1` - *sampling_protocol*
- `uniform/3` - *sampling_protocol*
- `uninstall/0` - *packs*
- `uninstall/1` - *packs*
- `uninstall/2` - *packs*
- `union/3` - *atms_environment_protocol*, *interval_relation_set_protocol*, *setp*, *two3tree*, *unweighted_directed_graph(Dictionary)*
- `union/4` - *setp*, *union_find_protocol*
- `union_all/3` - *union_find_protocol*
- `unique_min_cut/4` - *max_flow_protocol*
- `universal/1` - *interval_relation_set_protocol*
- `unix_seconds/2` - *ccsds_time_code_protocol*
- `unix_to_date_time/2` - *datep*
- `unpin/0` - *packs_common*
- `unpin/1` - *packs_common*
- `unregister/2` - *http_websocket_service_registry*
- `unsat_core_name/2` - *omt_protocol*, *smt_protocol*
- `unsubscribe/3` - *stomp*
- `unsubscribe/4` - *mqtt*
- `unzip/2` - *zipperp*
- `update/0` - *doclet*, *packs*, *registries*
- `update/1` - *observer*, *packs*, *registries*
- `update/2` - *packs*, *registries*
- `update/3` - *baseline_forecasting*, *datalog_protocol*, *dictionaryp*, *exponential_smoothing*, *intermittent_demand_forecasting*, *knn_forecasting*, *packs*, *time_series_regression*
- `update/4` - *dictionaryp*, *trie(Representation)*
- `update/5` - *dictionaryp*, *lru_cache_protocol*, *trie(Representation)*
- `update_fecf/2` - *ccsds_aos_frames(FrameLength,InsertZoneLength,HasOCF,HasFECF)*, *ccsds_frame_protocol*, *ccsds_frames*, *ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF)*, *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- `update_hash_state/3` - *hash_state_protocol*
- `update_in/4` - *nested_dictionary_protocol*
- `update_in/5` - *nested_dictionary_protocol*
- `update_ratings/3` - *bm25_recommender*, *jaccard_recommender*, *tfidf_recommender*
- `upper_case/2` - *text_normalizer_protocol*
- `upper_triangular_part/2` - *linear_algebra_protocol*
- `upper_triangular_part/3` - *linear_algebra_protocol*
- `user_data/2` - *ccsds_packets(SecondaryHeaderLength)*
- `userinfo/4` - *open_id*, *open_id_client*
- `utc_date_time_to_instant/2` - *time_scales_protocol*
- `utc_to_local/3` - *datep*

- `utc_to_local_tz/3` - *dates_tz_protocol*
- `utm_crs/2` - *crs_projections_protocol*
- `utm_zone/2` - *crs_projections_protocol*
- `uuid_max/1` - *uuid_protocol*
- `uuid_nil/1` - *uuid_protocol*
- `uuid_null/1` - *uuid_protocol*
- `uuid_v1/2` - *uuid_protocol*
- `uuid_v1/3` - *uuid_protocol*
- `uuid_v3/3` - *uuid_protocol*
- `uuid_v4/1` - *uuid_protocol*
- `uuid_v5/3` - *uuid_protocol*
- `uuid_v7/1` - *uuid_protocol*
- `uuid_v7/2` - *uuid_protocol*
- `valid/0` - *command_line_option*
- `valid/1` - *ccsds_frame_protocol*, *ccsds_frames*, *ccsds_packet_services*, *ccsds_time_code_protocol*, *ccsds_time_codes*, *ieee_754_protocol*, *interval_protocol*, *statisticsp*, *termp*, *typeid_protocol*, *uri_template(Representation)*, *url(Representation)*, *validation(Validation)*, *varlistp*
- `valid/2` - *type*
- `valid/3` - *datep*, *timep*
- `valid_anomaly_detector/1` - *anomaly_detector_protocol*
- `valid_association_rule_miner/1` - *association_rule_miner_protocol*
- `valid_channel_reassembly_state/1` - *ccsds_packet_services*
- `valid_classifier/1` - *classifier_protocol*
- `valid_clusterer/1` - *clusterer_protocol*
- `valid_column_type/1` - *souffle_program*
- `valid_conversion/3` - *time_scales_protocol*
- `valid_cookie/1` - *http_cookies(Representation)*
- `valid_coordinate/1` - *geospatial_protocol*
- `valid_coordinate/2` - *crs_projections_protocol*
- `valid_crs/1` - *crs_projections_protocol*
- `valid_cut/5` - *flow_network_protocol*
- `valid_date/3` - *iso8601*
- `valid_date_time/1` - *datep*
- `valid_descriptor/1` - *ccsds_time_fields*
- `valid_dimension_reducer/1` - *dimension_reducer_protocol*
- `valid_discontinuity_policy/1` - *ccsds_link_profiles*, *ccsds_packet_services*, *ccsds_tc_services*
- `valid_flow/3` - *flow_network_protocol*
- `valid_forecaster/1` - *forecaster_protocol*
- `valid_geohash/1` - *geohash_protocol*
- `valid_identifier/1` - *asp_program*, *minizinc_program*, *souffle_program*
- `valid_instant/1` - *time_scales_protocol*
- `valid_map_dispatch/1` - *ccsds_tc_services*
- `valid_min_cost_flow/4` - *flow_network_protocol*
- `valid_min_cost_flow/5` - *flow_network_protocol*
- `valid_model/1` - *ripple_down_rules_common_protocol*
- `valid_option/1` - *options_protocol*
- `valid_options/1` - *options_protocol*
- `valid_pattern_miner/1` - *pattern_miner_protocol*
- `valid_prefix/1` - *typeid_protocol*

- valid_profile/1 - *ccsds_link_profiles*
- valid_ranker/1 - *ranker_protocol*
- valid_reassembled_segment/1 - *ccsds_tc_services*
- valid_reassembly_state/1 - *ccsds_link_profiles*, *ccsds_packet_services*, *ccsds_tc_services*
- valid_recommender/1 - *recommender_protocol*
- valid_regressor/1 - *regressor_protocol*
- valid_result/1 - *problog_output*
- valid_scale/1 - *time_scales_protocol*
- valid_segment/1 - *ccsds_tc_services*
- valid_selector/1 - *feature_selector_protocol*
- valid_set_cookie/1 - *http_cookies(Representation)*
- valid_st_flow/5 - *flow_network_protocol*
- valid_suffix/1 - *typeid_protocol*
- valid_until_date/1 - *application_protocol*
- valid_vectorizer/1 - *text_vectorizer_protocol*
- validate/1 - *flags_validator*, *geojson_protocol*, *gpx_protocol*, *jwt_jwks*, *paseto_keys*, *wkt_wkb_protocol*
- validate/2 - *geojson_protocol*, *gpx_protocol*, *json_schema_protocol*, *wkt_wkb_protocol*
- validate/3 - *json_schema_protocol*
- validate_claim/3 - *jwt*, *jwt_claims*, *paseto*, *paseto_claims*
- validate_claims/3 - *jwt*, *jwt_claims*, *paseto*, *paseto_claims*
- validate_document/1 - *open_api*
- validate_document/2 - *open_api*
- validate_grouped_dataset/1 - *ranking_dataset_common*
- validate_grouped_dataset/2 - *ranking_dataset_common*
- validate_header/1 - *jwt_jwa*
- validate_http_request/3 - *open_api*
- validate_http_request/4 - *open_api*
- validate_http_response/3 - *open_api*
- validate_http_response/4 - *open_api*
- validate_key_set/1 - *paseto*
- validate_multiplayer_dataset/1 - *ranking_dataset_common*
- validate_multiplayer_dataset/2 - *ranking_dataset_common*
- validate_pairwise_dataset/1 - *ranking_dataset_common*
- validate_pairwise_dataset/2 - *ranking_dataset_common*
- validate_pairwise_measurement_dataset/1 - *ranking_dataset_common*
- validate_pairwise_measurement_dataset/2 - *ranking_dataset_common*
- validate_request/3 - *open_api*
- validate_request/4 - *open_api*
- validate_response/3 - *open_api*
- validate_response/4 - *open_api*
- validate_temporal_pairwise_dataset/1 - *ranking_dataset_common*
- validate_temporal_pairwise_dataset/2 - *ranking_dataset_common*
- valids/2 - *validated*
- value/1 - *counter*
- value/3 - *pairs*
- value_reference/2 - *java_utils_protocol*
- values/2 - *dictionaryp*, *pairs*, *trie(Representation)*
- values/3 - *trie(Representation)*
- variable/4 - *linear_programming_protocol*

- variable/6 - *linear_programming_protocol*
- variable_value/3 - *linear_programming_protocol*
- variables/2 - *sat_program*, *termp*, *uri_template(Representation)*
- variance/2 - *statisticsp*
- variant/2 - *lgtunit*, *termp*
- varnumbers/2 - *termp*
- varnumbers/3 - *termp*
- vector_norm/3 - *linear_algebra_protocol*
- vector_value/3 - *linear_algebra_protocol*
- vectorizer_options/2 - *text_vectorizer_protocol*
- verified_accounts/2 - *gravatar*
- verify/3 - *http_authenticate_verifier_protocol*, *http_oauth_verifier_protocol*
- verify/4 - *jwt*, *paseto*
- verify/5 - *jwt*, *jwt_jws*, *jwt_openssl*, *open_id_openssl*, *paseto*
- verify_commands_availability/0 - *packs_common*
- verify_fecf/1 - *ccsds_aos_frames(FrameLength,InsertZoneLength,HasOCF,HasFECF)*, *ccsds_frame_protocol*, *ccsds_frames*, *ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF)*, *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- verify_id_token/4 - *open_id*, *open_id_jwks_cache*
- verify_id_token/5 - *open_id*, *open_id_jwt*
- verify_password_hash/2 - *crypto*
- verify_signature/5 - *jwt*
- version/1 - *application_protocol*, *http_request_protocol*, *http_response_protocol*
- version/2 - *ccsds_frame_protocol*, *ccsds_frames*, *ccsds_packets(SecondaryHeaderLength)*, *http_core*, *memcached*
- version/6 - *pack_protocol*
- versions/3 - *packs*
- vertices/2 - *flow_network_protocol*, *graph_protocol*
- vincenty_distance/3 - *geospatial_protocol*
- virtual_channel_frame_count/2 - *ccsds_aos_frames(FrameLength,InsertZoneLength,HasOCF,HasFECF)*, *ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF)*
- virtual_channel_id/2 - *ccsds_frame_protocol*, *ccsds_frames*
- vocabulary/2 - *text_vectorizer_protocol*
- void/1 - *java_utils_protocol*
- void_element/1 - *html*
- von_mises/3 - *sampling_protocol*
- wait/2 - *process*
- wait/3 - *process*
- wald/3 - *sampling_protocol*
- wall_time/1 - *osp*
- warning/1 - *linter_reporter*
- warnings/1 - *linter_reporter*
- weakly_connected_components/2 - *directed_graph_protocol*
- websocket_accept/2 - *http_websocket_handshake*
- websocket_opening_key/1 - *http_websocket_handshake*
- week_of_year_iso/2 - *datep*
- weekday/2 - *datep*
- weibull/3 - *sampling_protocol*
- weight_variances/2 - *bayesian_ridge_regression*
- weighted_mean/3 - *statisticsp*

- welcome/0 - *shell*
- well_known_url/2 - *http_oauth_metadata*
- when/2 - *coroutining*
- whiledo/2 - *loopp*
- white_space//0 - *blank_grammars(Format)*
- white_spaces//0 - *blank_grammars(Format)*
- why/3 - *atms_protocol*
- window_interval/3 - *stn_protocol*
- window_relation/4 - *stn_protocol*
- with_connection/1 - *amqp_pool*
- with_output_to/2 - *term_io_protocol*
- within_distance/4 - *geospatial_protocol*
- without//2 - *sequence_grammars*
- wneighbors/3 - *weighted_graph_protocol*
- word32_hex/2 - *hash_common_32*
- word64_hex/2 - *hash_common_64*
- working_directory/1 - *osp*
- working_tree_clean/1 - *git_protocol*
- working_tree_status/2 - *git_protocol*
- write_file/3 - *csv_protocol*, *tsv_protocol*
- write_frame/2 - *http_websocket_frames*
- write_framed_message/2 - *json_rpc*
- write_max_depth/1 - *debuggerp*
- write_message/2 - *http_websocket_messages(TextRepresentation)*, *http_websocket_session(Role,TextRepresentation)*, *json_rpc*
- write_message/3 - *http_websocket_session(Role,TextRepresentation)*
- write_message/4 - *http_websocket_session(Role,TextRepresentation)*
- write_message/5 - *http_websocket_session(Role,TextRepresentation)*
- write_request/2 - *http_client_core*
- write_response/2 - *http_server_core*
- write_stream/3 - *csv_protocol*, *tsv_protocol*
- write_term_to_atom/3 - *term_io_protocol*
- write_term_to_chars/3 - *term_io_protocol*
- write_term_to_chars/4 - *term_io_protocol*
- write_term_to_codes/3 - *term_io_protocol*
- write_term_to_codes/4 - *term_io_protocol*
- write_to_atom/2 - *term_io_protocol*
- write_to_chars/2 - *term_io_protocol*
- write_to_codes/2 - *term_io_protocol*
- x25519_keypair/2 - *x25519*
- x25519_public_key/2 - *x25519*
- x25519_shared_secret/3 - *x25519*
- xchacha20/4 - *xchacha20_poly1305*
- xchacha20_poly1305_decrypt/5 - *xchacha20_poly1305*
- xchacha20_poly1305_encrypt/5 - *xchacha20_poly1305*
- xchacha20_subkey_and_nonce/4 - *xchacha20_poly1305*
- xml_to_gpx/2 - *gpx_protocol*
- xml_whatwg_named_entity/2 - *xml_whatwg_entities*
- xor64/3 - *hash_common_64*

- `year/2` - *ccsds_ccs(CalendarVariant,FractionOctets)*
- `z_normalization/2` - *statisticsp*
- `zadd/5` - *redis*
- `zcard/3` - *redis*
- `zero_or_more//0` - *sequence_grammars*
- `zero_or_more//1` - *sequence_grammars*
- `zero_or_more//2` - *sequence_grammars*
- `zip/2` - *zipperp*
- `zip/3` - *expected(Expected)*, *optional(Optional)*, *validation(Validation)*, *zipperp*
- `zip_at_index/4` - *zlist*
- `zone/3` - *tzif_protocol*
- `zone_id_kind/2` - *tzif_zone_ids*
- `zones/1` - *tzif_protocol*
- `zones/2` - *tzif_protocol*
- `zrange/5` - *redis*
- `zrank/4` - *redis*
- `zrem/4` - *redis*
- `zscore/4` - *redis*

INDEX

Symbols

(/)/2, 1202
(//)/2, 1202
(<)/2, 3592
(<=)/2, 101, 2066
(=:=)/2, 3593
(=<)/2, 3592
(=>)/2, 101, 2066
(=\=)/2, 3593
(>)/2, 3592
(>=)/2, 3593
=~= / 2, 2014, 3633

A

a_star_interpreter(W), 3781
abbreviation/1, 3519
abbreviation/2, 3682
abbreviation/3, 3682
abbreviation/4, 3681
abort_transaction/3, 3240
abs_max/2, 463
abs_sum/2, 464
absolute_file_name/2, 2597
absolute_url_context/2, 1319
absorb/2, 1140
acc_info/5, 803
acc_info/7, 803
accept/3, 1509, 1531
accept/4, 1510, 1531
accept_hook_/1, 2082
accept_loop_stopped_/0, 2083
accept_sse/3, 1456
accept_websocket/3, 1454
accumulate_feature_statistics/5, 2983
accumulate_forecast_error/4, 3468
accumulate_item_scores/3, 2887
accumulate_ridge_system/5, 2986
ack/3, 3238
acquire/1, 25
activate_debug_handler/1, 489
activate_monitor/0, 818
active_collection_mode_/1, 1981
active_debug_handler/1, 488
active_debug_handler_/1, 496
active_diagnostic_/1, 1980
active_feature_count/2, 2984
active_preflight_issue_/1, 1981
active_scores/3, 1959
active_spec_object_/1, 2229
active_transport_object_/1, 2204
active_worker_/3, 1569
adaptive_boosting_classifier, 1
adaptive_simpson_quadrature(Function), 2450
add/1, 2666
add/2, 2665
add/3, 2664, 3596
add/5, 2248
add32/3, 1145
add32/4, 1145
add32/5, 1145
add64/3, 1149
add_authentication_info/4, 1353
add_constraint/5, 3119
add_constraint/6, 3119
add_constraints/3, 3119
add_constraints/4, 3120
add_digest_authentication_info/3, 1357
add_directive_/2, 3821
add_directive_/3, 3821
add_directive_before_entity_/2, 3821
add_duration/3, 583
add_edge/4, 1091
add_edge/5, 1101
add_edges/3, 1072
add_htmx_response_headers/3, 1372
add_library_documentation_url/4, 722
add_link_options/3, 676
add_matrices/3, 2098
add_node_zoom_option/4, 677
add_outer_product/4, 2986
add_response_headers/4, 1333, 1370
add_rule/3, 573
add_scaled_outer_product/4, 2117
add_scaled_vector/4, 2092
add_server_session_response/3, 1475
add_squared_vector/3, 2983
add_time_points/3, 3116
add_type/2, 2285
add_type/3, 2285
add_vectors/3, 2091

add_vertex/3, 1071
add_vertices/3, 1071
addDependent/1, 634
adjacency/3, 933
adjacent/3, 989
adjacent_bits/4, 995
adjacent_int/4, 993
aes128, 145
aes192, 146
aes256, 147
aes_common(KeySize,Nk,Nr), 148
af_language_profile, 1873
after/2, 587, 1669
after/3, 497
after_event_registry, 809
agglomerative_clusterer, 3
aggregate_body_predicate/2, 570
aggregate_resources/2, 1246
alias/1, 318
aligned_residual_fits/5, 3477
all/0, 401, 603, 1991
all/1, 401, 603, 1991
all_files/0, 668, 688
all_files/1, 668, 688
all_libraries/0, 663, 683
all_libraries/1, 662, 683
all_pairs_min_paths/2, 1077
all_pairs_min_predecessors/2, 1077
all_score/1, 403
allowed_algorithm/2, 1836
along_track_distance/4, 1014
alternating_subsequence/2, 3348
alternating_subsequences/2, 3348
amqp, 4
amqp_pool, 23
ancestor/1, 1216
ancestor/4, 410
ancestors/1, 1216
anchor_series/2, 3431
anchors/4, 900
and64/3, 1151
annotate_htmx_request/2, 1372
annotate_server_session_request/2, 1475
anomaly_dataset_protocol, 28
anomaly_detector_common, 30
anomaly_detector_diagnostics_data/2, 31
anomaly_detector_export_template/2, 31
anomaly_detector_options/2, 37
anomaly_detector_protocol, 34
anomaly_detector_term_template/2, 32
anova_f_score, 852
ant_colony(Problem), 40
ant_colony(Problem,Random), 41
ant_colony_problem_protocol, 44
api_info/1, 1428, 2523
apid/2, 279
apis/0, 1200
apis/1, 1200
append/2, 3606
append/3, 2249, 2792, 3606, 3663
append/4, 2948
append_query/3, 2552
append_tls_transport/3, 2317
application_common, 47
application_protocol, 48
application_resource_template_uri/2, 2220
apply/2, 3840
apply/3, 1780
apply/4, 752
apply_float_sign/3, 1609
apply_session_read/6, 1595
approximately_equal/2, 2012, 3631
approximately_equal/3, 2013, 3631
apr1/3, 527
apriori_pattern_miner, 58
ar_language_profile, 1874
arbitrary, 59
arbitrary/1, 61
arbitrary/2, 61
archive/1, 2675
arcs/2, 927
arithmetic_mean/2, 3219
arithmetic_mean/5, 3211
arithmetic_operator_replacement(Entity,Predicate,Cl
2371
arrangement/3, 67
arrangement/4, 68
arrangement_index/4, 70
arrangement_index/5, 70
arrangements, 65
arrangements/3, 67
arrangements/4, 67
arrangements_protocol, 66
array_list/2, 1753
array_to_list/2, 1752
array_to_terms/2, 1752
array_to_terms/3, 1752
articulation_points/2, 1083
as_curly_bracketed/2, 744, 2441
as_curly_bracketed_impl/2, 761
as_curly_bracketed_impl/3, 761
as_deque/2, 641
as_dictionary/2, 743
as_difflist/2, 3603
as_heap/2, 1190
as_list/2, 640, 743, 1190, 2189, 2793, 3020, 3102, 3546, 3596
as_list/3, 3547
as_nested_dictionary/2, 2440
as_set/2, 3102
as_trie/2, 3546
ask_question/5, 487

asp, 74
asp_expansion, 76
asp_output, 77
asp_problem_protocol, 79
asp_process, 80
asp_program, 83
asp_protocol, 86
asp_statement/2, 80
asp_symbols, 93
aspiration/3, 3361
assert_fact/1, 574
assertion/1, 96, 2008
assertion/2, 96, 2009
assertions, 94
assertions(Mode), 95
assertions_messages, 97
assignable/1, 100
assignable/2, 100
assignvars, 98
assignvarsp, 99
association_rule_miner, 102
association_rule_miner_common, 103
association_rule_miner_options/2, 107
association_rule_miner_protocol, 105
assumption/2, 122
assumptions/2, 123
atms, 110
atms_bitset_environment, 111
atms_environment_protocol, 112
atms_ordered_list_environment, 115
atms_protocol, 116
atms_segmented_bitset_environment, 126
atom, 3578
atom_bytes/2, 1834, 2550
atom_list/1, 1834
atom_string/2, 3324
atomic, 3580
atomics_to_string/2, 3327
atomics_to_string/3, 3328
attach_sse_stream/1, 2231
attribute_occurrences/4, 2978
attribute_value/3, 375, 781
attribute_values/2, 29, 364, 386, 794, 862, 2964
augment/4, 934
augment_arc/4, 934
augment_rows/3, 2987
augmented_lagrangian(Problem,InnerSolver), 456
authenticate_request/4, 1386
authenticated_channel, 514
authenticated_channel_accept/5, 516
authenticated_channel_decrypt/5, 519
authenticated_channel_encrypt/5, 518
authenticated_channel_finalize/3, 517
authenticated_channel_initiate/4, 515
authentication_info/2, 1345
authentication_scheme_codes/3, 1285
authentication_token_codes/1, 1286
authorization/2, 1250, 1345, 1383
authorization_code/4, 2528, 2569
authorization_response/3, 2527, 2569
authorization_url/5, 2527, 2567
authorize_basic_auth_request/2, 1259
authorize_digest_auth_request/2, 1356
authorize_oauth_request/2, 1408
authorize_request/3, 1387
authorize_request/6, 1350
authorize_routed_request/2, 1431
auto_dispatch_tool/5, 2221
automation_report, 1992
auxiliary_predicate_counter_/1, 2040
available/0, 2622
available/1, 27, 2621
available/2, 2621
avatar_alt_text/2, 1112
avatar_url/2, 1112
average/2, 3637
average_deviation/3, 3221
avltree, 739
avro, 127
azure_list_command/4, 1724
azure_search_command/6, 1724
azure_states_query/1, 1725
azure_wiql/3, 1725

B

back_substitution/2, 2989
backend_adapter_hook, 1229
backend_random, 2795
backward_substitution/3, 2120
barrier_sub_problem(Problem,OuterIteration,StartPo
457
barycentric_interpolator, 2451
barzilai_borwein(Problem), 2146
base32, 130
base58, 132
base64, 134
base64url, 136
base64url_atom_bytes/2, 1833, 2551
base64url_no_padding, 138
base85, 140
base_/2, 984
base_dimension_reducer_diagnostics/6, 782
base_forecaster_diagnostics/5, 3465
base_recommender_diagnostics/5, 2921
base_regressor_diagnostics/6, 2970
base_selector_diagnostics/5, 882
baseline_coverage_/4, 2390
baseline_forecasting, 142
baseline_training_examples/3, 33
basic_ack/3, 15
basic_cancel/3, 14

basic_consume/3, 14
basic_get/3, 15
basic_nack/3, 16
basic_publish/4, 13
basic_qos/2, 17
basic_recover/2, 17
basic_reject/3, 16
basis_initial_vectors/3, 784
basis_vector/3, 783, 2094
bayesian_ridge_regression, 143
bbox_contains/2, 1020
bbox_contains_polygon/2, 1022
bbox_expand/3, 1021
bbox_from_coordinates/2, 1022
bbox_intersects/2, 1020
bbox_intersects_polygon/2, 1021
bbox_intersects_polyline/2, 1022
bbox_overlaps/2, 1020
bbox_union/3, 1021
bcrypt/4, 528
before/2, 587, 1668
before/3, 497
before_event_registry, 810
begin/0, 574
begin/3, 1478
begin_transaction/3, 3239
bell/2, 3628
bench_goal/1, 3791
benchmark/2, 2011
benchmark/3, 2011
benchmark/4, 2012
benchmark_generators, 3782
benchmark_reified/3, 2011
bernoulli/2, 2818
best_effort_protocol_error_close/3, 1596
best_first, 3783
best_supported_spec/3, 2211
beta/3, 2819, 3736
beta_density/4, 3737
beta_distribution/4, 3739
beta_log_density/4, 3738
beta_quantile/4, 3739
beta_samples/4, 3737
better_candidate/3, 1645
better_value/3, 2160
between/3, 2807, 3600
between/4, 3598
bfgs(Problem), 2148
bfs/5, 933
bfs/6, 933
bfs_interpreter, 3785
bg_language_profile, 1875
bic/5, 1717
bigrams/2, 2428
binary_body/2, 3006
binary_contents/2, 1738
binary_descriptor/2, 1738
binary_file_assertion/3, 2037
binary_heap(Order), 1183
binary_heap_max, 1184
binary_heap_min, 1185
binary_input_assertion/2, 2024
binary_input_assertion/3, 2024
binary_output_assertion/2, 2032
binary_output_assertion/3, 2032
binary_output_contents/1, 2033
binary_output_contents/2, 2033
binary_pairs/2, 1739
binomial/3, 2818, 3629
bintree, 740
bisection_root_finder(Function), 2452
bit//1, 1053
bits//1, 1053
bits_fields/5, 1606
bits_finite_binary_rational/4, 1607
bits_nan_kind/2, 1607
bits_nan_payload/2, 1607
blake2_block_size/1, 1120
blake2_block_words/2, 1121
blake2_core(Key,DigestSize), 1118
blake2_finalize/4, 1122
blake2_g/10, 1121
blake2_initial_state/2, 1120
blake2_max_digest_size/1, 1119
blake2_max_key_size/1, 1119
blake2_round_count/1, 1120
blake2_state_bytes/2, 1122
blake2_working_vector/4, 1121
blake2b, 1123
blake2b(Key,DigestSize), 1124
blake2s, 1125
blake2s(Key,DigestSize), 1126
blank//0, 1048
blank_grammars(Format), 1045
blanks//0, 1048
block_cipher_modes_common, 149
block_cipher_prepared_key_protocol, 154
block_cipher_protocol, 157
block_maximum/5, 1958
block_size/1, 159, 1154
blocks/3, 153
bm25_recommender, 169
bn_language_profile, 1876
body/1, 1323, 1326
body/2, 1303
body/3, 1374
body_goal_negation(Entity,Predicate,ClauseIndex,Occ
2372
body_pred/1, 2275
body_pred_call/2, 2278
borda_ranker, 186
bottleneck/3, 935

bounded_similarity/2, 2933
bounding_box/2, 988
bounding_box/3, 1026
bounding_box_bits/3, 995
bounding_box_int/3, 992
bounds/4, 3123
br_language_profile, 1877
bradley_terry_ranker, 188
branch/2, 1030
breadth_first_order/3, 1074
brent_root_finder(Function), 2454
bridges/2, 1083
broadcast/2, 1582
broadcast_except/3, 1583
bson, 190
bson(StringRepresentation), 191
build/4, 951
build/6, 957
build_auto_search_index/3, 388
build_capabilities/2, 2210
build_dataset_model/5, 2850
build_direct_strengths/5, 2834
build_encoders/4, 376, 781
build_grid_index/3, 388
build_inner/3, 464
build_linear_encoders/4, 356, 2981
build_metric_tree/3, 388
build_ridge_system/5, 2985
build_score_ranker/5, 2885
build_vectors/5, 3424
built_date/1, 53
built_in_directive/4, 1203
built_in_flag/2, 910
built_in_method/4, 1204
built_in_non_terminal/4, 1205
built_in_predicate/4, 1204
bup_interpreter, 3786
bypass_flag/2, 229
byte_count/1, 1615, 1619
byte_list/1, 2713
byte_order, 193
bytes_atom/2, 1834, 2551
bytes_hex/2, 1144
bytes_to_codes/2, 319
bytes_to_integer/3, 194
bytes_to_integer/5, 194
bytes_to_signed_integer/3, 195
bytes_to_signed_integer/5, 196

C

c/1, 3789
c45_classifier, 196
ca_language_profile, 1878
cache/1, 3675
cache_policy/4, 2193
cache_source/1, 3677
cached_jwks/3, 2532, 2556
cached_jwks_/3, 2559
cached_run_/5, 1980
cached_tzif/1, 3677
cached_tzif_/2, 3672
calendar_month/3, 1692
call_with_timeout/2, 3499
call_with_timeout/3, 3499
callable, 3581
caller_diagram, 652
caller_diagram(Format), 653
can_precede/3, 3124
can_precede_or_equal/3, 3125
candidate_key/2, 1645
canonical_bic/2, 1717
canonical_contents/3, 2903
canonical_iban/2, 1695
canonical_nan_bits/1, 1607
canonical_query/2, 3060
canonical_request/6, 3060
canonical_uri/4, 3028
canonical_uri/5, 3029
capabilities/1, 2241
capability/2, 2495, 3156
capacity/2, 2185
capacity_scaling, 915
capacity_scaling(Dictionary), 916
captured_mutated_terms_/5, 2390
capturing_mutated_terms_/0, 2390
cartesian_product/2, 202
cartesian_product/3, 202
cartesian_products, 200
cartesian_products_protocol, 201
cartesian_tuple/2, 202
cartesian_tuple/3, 203
cartesian_tuple_index/3, 206
cartesian_tuple_index/4, 206
cas/6, 2250
case_conversion/3, 3385
case_fold/2, 3391
cat/2, 2573
catalan/2, 3627
catalog_score/5, 1737, 3422
catalog_score/6, 184
categorical_pairs/4, 868
caverphone/2, 3322
caverphone_match/2, 3322
cbc, 159
cbor, 209
cbor(StringRepresentation), 210
cc_metric, 395
ccsds_aos_frames(FrameLength,InsertZoneLength,HasO
212
ccsds_ccs(CalendarVariant,FractionOctets),
298

ccsds_cds(DaySegmentOctets,SubmillisecondOctets,Epoch), 300
ccsds_cuc(CoarseOctets,FineOctets,Epoch), 302
ccsds_frame_protocol, 216
ccsds_frames, 221
ccsds_frames_types, 227
ccsds_link_profiles, 236
ccsds_packet_services, 247
ccsds_packetization, 267
ccsds_packets, 275
ccsds_packets(SecondaryHeaderLength), 276
ccsds_packets_types, 282
ccsds_tc_frames(FrameLength,SegmentHeaderLength,HasFECF), 228
ccsds_tc_services, 283
ccsds_time_code_protocol, 304
ccsds_time_codes, 308
ccsds_time_codes_types, 310
ccsds_time_fields, 311
ccsds_tm_frames(FrameLength,SecondaryHeaderLength,HasFECF), 231
cdata_generation//1, 3829
cell_dimensions/3, 989
center/3, 872, 2935
center_gram_matrix/4, 2097
center_gram_vector/4, 2098
centered_normalized_values/2, 2934
centered_values/2, 871
central_moment/3, 3227
central_moment_sum/7, 3213
chacha20_block/4, 542
chacha20_encrypt/5, 542
challenge/2, 1250, 1345, 1383
change_directory/1, 2601
changed/0, 633
changed/1, 633
channel_close/1, 8
channel_close/3, 8
channel_open/3, 8
character, 3582
character_data_format/3, 3828
character_n_grams/3, 2426
character_n_grams/4, 2426
character_set, 314
character_set_protocol, 317
characterp, 3583
chebyshev_distance/3, 3640
chebyshev_norm/2, 3639
check/0, 448
check/1, 3653, 3670
check/2, 3660
check/3, 3659
check_aligned/2, 152
check_anomaly_detector/1, 36
check_argument/3, 2460
check_association_rule_miner/1, 107
check_attribute_bindings/2, 375, 2977
check_attribute_declarations/1, 2976
check_binary_file/2, 2036
check_binary_input/1, 2023
check_binary_input/2, 2023
check_binary_output/1, 2032
check_binary_output/2, 2031
check_block/2, 152
check_bounds/1, 2161
check_bounds/2, 2473
check_bracket/3, 2480
check_bytes/1, 151
check_catalog_additions/2, 1737, 3420
check_catalog_entries/1, 178
check_catalog_identifiers/1, 177
check_catalog_kind/3, 179
check_catalog_replacements/2, 1736, 3419
check_categorical_value/3, 2980
check_classifier/1, 360
check_count/2, 378
check_clusterer/1, 382
check_complete_examples/2, 355
check_complete_examples_nonvar/2, 355
check_complete_pairs/2, 896
check_component_count/3, 778
check_content_kind/3, 182, 3419
check_continuous_attributes/1, 373, 780
check_conversion/3, 3452
check_convert/4, 3454
check_declarations/1, 906, 1963
check_declared_attribute_bindings/2, 2977
check_dimension_reducer/1, 790
check_encoded_attribute_bindings/2, 378
check_ensemble/2, 2
check_example_attributes/2, 375, 781
check_example_values/2, 374, 780
check_examples/2, 354, 2969
check_examples/3, 374
check_examples_non_empty/2, 32, 354, 374, 780
check_existing_catalog_items/2, 179
check_expected_counts/3, 348
check_feature_values/2, 2168
check_finite_ratings/1, 186, 1739, 3423
check_forecast_horizon/1, 3471
check_forecaster/1, 3490
check_frequency/1, 3473
check_initial_value/3, 2466
check_instance_ids/2, 2168
check_instant/1, 3451
check_item_domain/1, 2731
check_lock_options/1, 2649
check_model/1, 3020
check_new_catalog_items/2, 180
check_no_duplicate_ratings/1, 2914
check_numeric_residuals/2, 3477

check_observation/1, 3462
check_offset/3, 3455
check_option/1, 2588
check_options/1, 2588
check_pattern_miner/1, 2735
check_point/2, 2162
check_problem/1, 2122
check_query_identifiers/2, 2915
check_ranker/1, 2861
check_rated_catalog/2, 2904
check_rating_scale/2, 178
check_rating_updates/3, 181, 1735, 3421
check_ratings/2, 2914
check_recommender/1, 2928
check_regressor/1, 2992
check_result/1, 2122
check_rule_options/1, 104
check_scale_ratings/3, 178
check_scoring_metric/1, 883
check_search_count/1, 1957
check_selector/1, 888
check_sequences/4, 3095
check_series/2, 3461
check_series/3, 3461
check_series_length/3, 3462
check_target/2, 901
check_text_file/2, 2035
check_text_file/3, 2035
check_text_input/1, 2020
check_text_input/2, 2020
check_text_output/1, 2027
check_text_output/2, 2027
check_text_output/3, 2026
check_transactions/4, 962
check_undeclared_attribute_bindings/2, 2977
check_usable_examples/1, 2329
check_vectorizer/1, 3410
checked_forecaster/2, 1649
checksum/2, 2448
chi_square_feature_selector, 347
chi_square_score, 853
chi_square_score(Discretization), 854
chi_square_yates_score, 855
chi_squared/2, 2816, 3729
chi_squared_density/3, 3730
chi_squared_distribution/3, 3731
chi_squared_log_density/3, 3731
chi_squared_quantile/3, 3732
chi_squared_samples/3, 3729
children/2, 991
cholesky_decomposition/2, 2118
chr_is/2, 3539
chr_next_state/1, 3543
chr_no_spy/1, 3540
chr_nospy/0, 3540
chr_notrace/0, 3540
chr_option/2, 3541
chr_option_allow_deep_guards/0, 3543
chr_option_optimization_level/1, 3542
chr_option_print_trace/0, 3541
chr_option_show_history/0, 3543
chr_option_show_id/0, 3543
chr_option_show_stack/0, 3542
chr_option_show_store/0, 3542
chr_option_trace_interactive/0, 3542
chr_rule_/1, 3544
chr_spy/1, 3540
chr_spy_point/1, 3544
chr_trace/0, 3539
chunk_size_line_size/2, 1317
circular_uniform_cartesian/3, 2823
circular_uniform_polar/3, 2823
claim/3, 1814, 1823, 2705, 2710
claims/2, 1814, 1826, 2704
class/1, 29, 364, 795, 1209
class_/5, 1994
class_hierarchy, 1207
class_hierarchyp, 1208
class_scatter/5, 868
class_values/1, 29, 364, 795
classes/1, 1209
classical_complete_seasonal_adjustment/5, 3483
classical_seasonal_adjustment/5, 3474
classifier_common, 349
classifier_diagnostics_data/2, 350
classifier_export_template/4, 350
classifier_options/2, 361
classifier_protocol, 358
classifier_term_template/2, 351
classify/2, 1612
classify/3, 3018
classify/4, 3019
classify_with_context/5, 3022
clause/5, 2761
clause_/5, 2762
clause_breakpoint_/2, 615
clause_location/6, 2761
clause_location_/6, 2762
clause_mutator_protocol, 2373
clauses_/1, 2375
clauses_reordering(Entity,Predicate,ClauseIndex,Oc
2374
clean/0, 2642, 2670
clean/1, 2641, 2670
clean/2, 2641, 3398
clean/3, 3398
clean_binary_input/0, 2025
clean_binary_output/0, 2033
clean_directory/1, 2037
clean_file/1, 2037
clean_text_input/0, 2021

clean_text_output/0, 2030
cleanup/0, 2016, 2227, 2237
clear/0, 573, 2558
clear/1, 1468, 2558
clear/2, 118, 2189
clear_cache/0, 3676
clear_dut1_override/0, 3440, 3449
clear_leap_seconds_override/0, 3439, 3448
client_capabilities_/1, 2211
client_connection_/3, 2081
client_connection_alias_/2, 2078
client_connection_input_/2, 2078
client_connection_output_/2, 2078
client_engine_/2, 2083
client_open/4, 3178
client_open/5, 3177
client_timeout_/1, 2079
clip_rating/3, 2916
clo_span_pattern_miner, 368
clockwise_polygon/2, 1019
clone/1, 2061, 2674
clone/3, 744
clone/4, 744
clone_impl/3, 758
clone_impl2/4, 758
cloning, 2061
close/1, 7, 1338, 1439, 1460, 1468, 1478, 1513, 1534, 1579, 3049, 3160
close/2, 1514, 1534, 3181
close/3, 7
close_client/1, 2071
close_connection/1, 1495
close_connection_pool/1, 1496
close_handshake_completed/1, 1596
close_listener/1, 1493
close_polygon/2, 1017
closed_input_stream/2, 2038
closed_output_stream/2, 2038
cluster/3, 381
cluster_probabilities/3, 967
clusterer_common, 369
clusterer_diagnostics_data/2, 379, 3202
clusterer_options/2, 383
clusterer_protocol, 380
clustering_dataset_protocol, 385
cmac, 390
cmac_protocol, 391
cnf_statement/2, 3074
coarse/2, 303
cobertura_report, 1993
code_challenge/2, 2566
code_metric, 396
code_metrics, 407
code_metrics_messages, 408
code_metrics_utilities, 409
code_verifier/2, 2566
codes_to_bytes/2, 319
coefficient_of_variation/2, 3222
cogc_metric, 414
collapse_flow/4, 952
collect/4, 766
collect_contents/3, 2902
collection_options_/1, 2143
colley_ranker, 435
combination/3, 439
combination/4, 440
combination_index/4, 442
combination_index/5, 443
combinations, 437
combinations/3, 439
combinations/4, 440
combinations_protocol, 438
command/2, 2657
command_arguments/3, 1726
command_line_arguments/1, 2610
command_line_option, 447
command_line_options, 451
command_text/4, 1721
commit/0, 574
commit_author/2, 1032
commit_date/2, 1032
commit_hash/2, 1032
commit_hash_abbreviated/2, 1033
commit_log/3, 1033
commit_message/2, 1033
commit_transaction/3, 3240
common_pairs/3, 2932
common_prefix/3, 991
common_subsequence/3, 3344
common_subsequences/3, 3344
compact/3, 1771
compact/4, 1828
company/2, 1113
compare_date_time/3, 588
compare_feature_scores/3, 884
compare_numeric_pairs/3, 874
comparingp, 3591
compatible_key/2, 1840
compile_aux_clauses/1, 493
compile_predicate_heads/4, 494
compile_predicate_indicators/3, 494
compiled_pred_call/2, 2278
complement/2, 1091
complete_pairs/3, 869
complete_regressor_options/2, 1961
complete_values/2, 870
completion/2, 1203
completion/4, 2194
completion_request/5, 2220
completion_result_to_json/2, 2221
completions/2, 1203
component_iteration_diagnostics/4, 783

compose/3, 1087, 1655, 1678, 3576
compound, 3594
compress/2, 991
compress_features/3, 2985
compress_rows/3, 2984
compress_vector/3, 2985
compute_ranks/3, 3215
condition/0, 2016
conditional_breakpoint_/3, 617
condorcet_victory_common, 2833
confirm_select/1, 19
conjugate_gradient(Problem), 2149
connect/1, 2946
connect/2, 2247
connect/3, 2246, 2310, 2946
connect/4, 6, 2310, 3163, 3236
connected_components/2, 1094, 1106
connection_alive/1, 7, 2311, 3164, 3236
connection_persistent/2, 1315
connection_pool_available_/2, 1422, 1489
connection_pool_config_/6, 1421, 1488
connection_pool_in_use_/2, 1422, 1489
connection_pool_seed_/1, 1421, 1488
connection_pool_stats/2, 1497
connection_streams/3, 1495
consensus_ranking/2, 1851
consensus_score/2, 1851
consistent/1, 1662, 3117
consistent/2, 125
console/1, 2947
constant_forecast/3, 3470
constant_time_equal/2, 534
constrained_optimization_problem_protocol, 458
constrained_optimization_solver, 461
constraint/5, 2126
constraints/2, 1665, 3117
constraints/3, 2127
containment_edge_/2, 658
contains/2, 1671
content_documents/2, 184
context_breakpoint_/4, 616
context_seed_/1, 1484
context_state_/2, 1485
context_summaries/2, 3527
contingency_counts/2, 869
contingency_min_expected_count/2, 869
contingency_score/3, 869
continuation_byte/1, 315
continuous_stats/5, 2974
contradiction/2, 123, 1660
contradiction_explanations/2, 1661
control//0, 1049
control_command_flag/2, 229
control_construct/4, 1204
controls//0, 1049
converse/2, 1655, 1678
convert/4, 3454
convert_case_codes/4, 3380
convert_zones/4, 595
convert_zones_with_resolution/5, 595
cookie_attribute_present/2, 1282
cookie_attribute_value/3, 1282
cookie_attribute_value/4, 1283
cookie_count/2, 1471
cookie_deletion/3, 1284
cookie_expiry/2, 1283
cookie_expiry/3, 1283
cookie_jar/2, 1338, 1461
cookies/2, 1471
cooling_schedule/3, 3134
coordinates_bounding_box/2, 1016
copeland_ranker, 477
copy_file/2, 2606
copy_object/5, 3040
copy_object/6, 3056
copy_object_response/2, 3064
core_messages, 479
coroutining, 499
correct_spelling_codes/3, 3379
correction_steps/4, 3433
correlation/3, 3226
correlation_score, 856
cosine_similarity, 2898
cosine_similarity/3, 1908, 2919, 3319
cosine_similarity/5, 1908
count/2, 1483, 2427
count/3, 2427
count_arrangements/3, 71
count_cartesian_product/2, 204
count_combinations/3, 443
count_derangements/2, 648
count_distinct_arrangements/3, 71
count_distinct_cartesian_product/2, 205
count_distinct_combinations/3, 443
count_distinct_derangements/2, 648
count_distinct_multisets/3, 2336
count_distinct_partitions/2, 1627, 2692
count_distinct_partitions/3, 1627, 2694
count_distinct_permutations/2, 2750
count_distinct_subsequences/3, 3345
count_frequencies/4, 3214
count_items/4, 963, 3096
count_length/3, 185
count_less_equal/6, 3215
count_multisets/3, 2335
count_numeric_targets/2, 1957
count_partial_derangements/3, 648
count_partitions/2, 1626, 2689
count_partitions/3, 1627, 2692
count_permutations/2, 2750
count_subsequences/2, 3349

counter, 3787
counter/2, 2063, 2419
counter_/2, 981, 2064, 2420
counterclockwise_polygon/2, 1019
counters, 2062
country/4, 1699, 1702
country_spec/3, 1697
coupling_metric, 415
covariance/3, 3226
covariance_matrix/2, 2117
cover/1, 2005
coverage_clause_mutator/0, 2400
coverage_entry_/4, 2407
coverage_file_/1, 2407
coverage_report, 1996
covered_/4, 2043
covering/3, 990
covering/4, 990
cpu_time/1, 591, 2608
cramers_v_bias_corrected_score, 857
cramers_v_bias_corrected_score(Discretization), 858
cramers_v_score, 859
cramers_v_score(Discretization), 860
crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength), 1127
crc32_reflected(Polynomial), 1128
crc32b, 1130
crc32bzip2, 1131
crc32c, 1132
crc32mpeg2, 1133
crc32posix, 1134
crc32q, 1135
create/3, 2782
create_arguments/7, 1727
create_assumption/4, 119
create_binary_file/2, 2034
create_contradiction/4, 119
create_multipart_upload_response/2, 3064
create_node/4, 118
create_text_file/2, 2034
create_text_file/3, 2034
created_response/4, 3007
creators/1, 52
crontab, 502
crontab(Representation), 503
cross_deviation_sum/6, 3212
cross_track_distance/4, 1014
crossover/4, 975
crossover_rate/4, 976
crs_dimensions/2, 510
crs_epsg/2, 510
crs_kind/2, 509
crs_name/2, 509
crs_projections, 506
crs_projections_protocol, 508
crs_units/2, 510
crypt/5, 164
crypt/6, 165
crypto, 520
cs_language_profile, 1879
csv, 545
csv(Header,Separator,IgnoreQuotes), 546
csv(Header,Separator,IgnoreQuotes,Comments), 547
csv_guess_questions, 548
csv_protocol, 549
ctr, 163
ctrf_output, 1998
ctrf_report, 1999
cubic_spline_interpolator, 2455
cuid2, 558
cuid2(Representation,Size,Alphabet), 559
cuid2_protocol, 560
curly_pairs_to_list/2, 1312
curly_to_pairs/2, 2224
currency/5, 1704, 1705
current/2, 1479, 3837
current_close_state/2, 1596
current_entity/1, 410
current_input/1, 3181
current_options/1, 2206, 2232
current_options_/1, 2217
current_predicate_clause_index_/2, 2396
current_predicate_directive_index_/2, 2397
current_prog/1, 3541
current_scope_directive_index_/2, 2397
current_url/2, 1365
current_url_abs_path/2, 1365
current_uses_directive_index_/2, 2398
custom_language_scripts/2, 1911
custom_profile/2, 1926
cusum_anomaly_detector, 562
cut/3, 1207
cut_capacity/3, 928
cycle/2, 1064, 1082
cycle_edge_/2, 679
cytoscapejs_graph_language, 655

D

d2_graph_language, 658
da_language_profile, 1880
damerau_levenshtein/3, 3315
data/0, 2759
data/1, 2759
data/2, 1480, 2448, 2760
data_field/2, 220, 224
data_length/2, 280
databasep, 3789
datalog, 563
datalog_protocol, 572
dataset_attributes/2, 32, 353, 371, 779, 2969

dataset_examples/2, 353, 879, 2969
dataset_examples/3, 879
dataset_protocol, 363
dataset_rating_scale/2, 2915
dataset_ratings/2, 2913
dataset_series/2, 3460
date, 578
date/4, 1682
date/5, 1682
date/6, 1683
date/7, 1684
date_string/3, 1685
date_time/7, 2608
date_time_string/3, 1688
date_time_to_unix/2, 582
datep, 579
dates_tz, 592
dates_tz_protocol, 593
day/2, 299
day_of_year/2, 299, 585
day_of_year_date/3, 585
daylight_saving_time/2, 3681
daylight_saving_time/3, 3681
daylight_saving_time/4, 3680
days/2, 301
days_in_month/3, 581
dbscan_clusterer, 596
de_language_profile, 1881
deactivate_debug_handler/0, 489
dead_code_scanner, 598
dead_code_scanner_messages, 605
debug/0, 621
debug_expansion(Mode), 3792
debug_handler/1, 488
debug_handler/3, 489
debug_messages, 606
debugger, 610
debugger_messages, 618
debuggerp, 619
debugging/0, 622
debugging/1, 622
debugging_/0, 612
decide/1, 851
decide/2, 851
declarations/3, 900
declared_attribute_names/2, 2978
declares_predicate/2, 411
decode/2, 988, 1800, 3206
decode/3, 1813, 1825, 2560, 3206
decode/5, 1813, 1825, 2291
decode/6, 2769
decode/7, 3140
decode/8, 78, 3071, 3185
decode_binary_data/3, 2322
decode_bits/3, 994
decode_body/4, 1288, 1300
decode_entities/2, 3394
decode_entities/3, 3395
decode_entities_codes/4, 3384
decode_exception/2, 1754
decode_exception/3, 1755
decode_field_value/3, 23
decode_fixed_header/5, 2327
decode_frame/2, 22
decode_int/3, 992
decode_message/3, 1557
decode_packet/2, 2315
decode_properties/3, 2324
decode_property/3, 2326
decode_property_list/2, 2325
decode_uint16/3, 2319
decode_uint32/3, 2320
decode_uint8/3, 2318
decode_url_spaces/2, 2659
decode_utf8_string/3, 2321
decode_utf8_string_pair/3, 2323
decode_varint/3, 2321
decompile_predicate_heads/4, 495
decompile_predicate_indicators/4, 495
decompose/3, 3576
decompose_file_name/3, 2598
decompose_file_name/4, 2598
decompose_flow/3, 929
decr/3, 2950
decr/4, 2253
decrby/4, 2951
decrement_counter/1, 2063
decrypt/4, 167, 2701
decrypt/5, 161, 2701
decrypt_block/3, 158
decrypt_padded/4, 168
decrypt_padded/5, 162
decrypt_prepared_block/3, 156
default/1, 450
default_atom_mutations, 2412
default_compound_mutations, 2413
default_float_mutations, 2414
default_integer_mutations, 2415
default_language_scripts/2, 1910
default_list_mutations, 2416
default_option/1, 2589, 3817
default_options/1, 2589, 3817
default_profile/2, 1926
default_text_normalization, 3377
default_workflow_hook, 1230
define_flag/1, 911
define_flag/2, 911
define_log_file/2, 2173
defined/4, 2664
defined_flag/6, 910
defined_flag_/6, 912
defines_predicate/2, 411

defines_predicate/3, 411
degree/3, 1094, 1106
del/3, 2951
del_monitors/0, 815
del_monitors/4, 815
del_spy_points/4, 819
delete/0, 2669
delete/1, 2669
delete/2, 2252, 2668
delete/3, 1266, 3102, 3607, 3663
delete/4, 745, 1189, 1267, 1341, 1463, 2188, 3550
delete_all_after/2, 3843
delete_all_after_and_unzip/2, 3843
delete_all_before/2, 3842
delete_all_before_and_unzip/2, 3843
delete_and_next/2, 3842
delete_and_previous/2, 3841
delete_and_unzip/2, 3842
delete_directory/1, 2600
delete_directory_and_contents/1, 2600
delete_directory_contents/1, 2600
delete_edge/4, 1091
delete_edge/5, 1102
delete_edges/3, 1073
delete_file/1, 2606
delete_impl/5, 759
delete_in/4, 2442
delete_matches/3, 3607
delete_max/4, 750
delete_min/4, 750
delete_object/4, 3039
delete_object/5, 3056
delete_prefix/3, 3551
delete_vertex/3, 1072
delete_vertices/3, 1072
demand_summary/3, 1648
demodb, 3793
dependent_/1, 635
dependents/1, 633, 2646
dependents/2, 2646
dependents/3, 2645
depth/2, 3651
depth_first_order/3, 1074
deque, 635
deque_protocol, 636
derangement/2, 644
derangement/3, 645
derangement_index/3, 647
derangement_index/4, 647
derangements, 641
derangements/2, 644
derangements/3, 644
derangements_protocol, 643
derivative/2, 2486
derivative/3, 2471
derivative/4, 2456
derive_aggregate_goals/1, 570
derive_aggregate_literal/2, 570
descendant/1, 1217
descendant_class/1, 1214
descendant_classes/1, 1214
descendant_instance/1, 1213
descendant_instances/1, 1213
descendants/1, 1217
describe/1, 2628, 2663
describe/2, 2628
description/1, 50, 2617, 2674
description/2, 1113
descriptor_document/2, 184
destination_point/4, 1013
destroy/0, 24
destroy/1, 1482
detach_sse_stream/0, 2231
detect/2, 1916
detect/3, 1916
detect/4, 1917
detect_all/2, 1918
detect_all/3, 1918
determinant/2, 2107
deterministic/1, 2007
deterministic/2, 2008
dfs_interpreter, 3794
diacritic_fold/2, 3385
diagnostic/2, 37, 106, 361, 383, 791, 889, 2734, 2860, 2929, 2993, 3412, 3491, 3530
diagnostic/3, 3530
diagnostic_position/2, 1651
diagnostic_rule/5, 3529
diagnostic_rules/1, 3529
diagnostic_target/1, 3529
diagnostics/2, 36, 106, 360, 382, 790, 889, 2734, 2860, 2928, 2993, 3411, 3490, 3531
diagnostics/3, 3530
diagnostics_breakdown/2, 3527
diagnostics_preflight/2, 3532
diagnostics_preflight/3, 3532
diagnostics_summary/2, 3531
diagnostics_summary/3, 3531
diagnostics_tool/5, 3529
diagonal_matrix/2, 2103
diagonal_matrix/3, 2103
diagonal_matrix/4, 2103
diagram(Format), 659
diagram_caption/3, 669
diagram_description/1, 669
diagram_name_suffix/1, 669
diagrams, 680
diagrams(Format), 681
dictionaryp, 742
dif, 767
dif/1, 500, 768
dif/2, 500, 768

difference_bounds/5, 3124
difference_norm/3, 2115
difference_series/2, 3466
differential_evolution(Problem), 769
differential_evolution(Problem,Random), 770
differential_evolution_problem_protocol, 773
difflist, 3595
digest/2, 1154
digest/4, 392, 1223
digest/5, 393, 1223
digest_size/1, 1154
digit//1, 1053
digits//1, 1053
dimension_reducer_common, 776
dimension_reducer_data/3, 778
dimension_reducer_diagnostics_data/2, 778
dimension_reducer_options/2, 791
dimension_reducer_protocol, 788
dimension_reduction_dataset_protocol, 793
dinic, 917
dinic(Dictionary), 918
directed_graph_common, 1060
directed_graph_protocol, 1061
direction_sign/2, 463
directive_mutator_protocol, 2376
directories/1, 1989, 3815
directories/2, 664, 685, 1988, 3814
directories/3, 664, 684
directory/1, 399, 601, 666, 687, 1989, 2382, 2654, 3814
directory/2, 398, 600, 666, 686, 1989, 2382, 2654, 3814
directory/3, 666, 686
directory_dependency_diagram, 689
directory_dependency_diagram(Format), 690
directory_diagram(Format), 692
directory_entity_/4, 1979
directory_exists/1, 2604
directory_files/2, 2603
directory_files/3, 2603
directory_load_diagram, 695
directory_load_diagram(Format), 696
directory_mutants/2, 2388
directory_mutants/3, 2389
directory_score/2, 403
dirichlet/2, 2363, 2822
dirichlet_density/3, 2365
dirichlet_log_density/3, 2365
dirichlet_samples/3, 2364
disable/0, 2141
disable/1, 608
disable/2, 609
disable_logging/1, 2175
disconnect/1, 2247, 2947, 3164
disconnect/2, 2311, 3236
discover/3, 2526, 2536
discovery_url/2, 2543
discrete_variables/2, 2123
disjoint/2, 3103
disjoint_sets/2, 3711
dispatch/3, 1453
dispatch_map_id/2, 285
dispatch_open_ai/2, 2510
dispatch_rest_endpoint/2, 3011
dispatch_service_units/2, 286
dispatch_service_units_by_map/2, 296
dispatch_service_units_by_map/3, 297
display_name/2, 1111
distance/4, 389, 1011, 3121
distance/5, 1012, 3122
distinct_arrangement/3, 68
distinct_arrangement/4, 69
distinct_arrangement_index/4, 72
distinct_arrangements/3, 68
distinct_arrangements/4, 69
distinct_cartesian_product/2, 203
distinct_cartesian_product/3, 204
distinct_cartesian_tuple/2, 203
distinct_cartesian_tuple/3, 204
distinct_cartesian_tuple_index/3, 207
distinct_combination/3, 441
distinct_combination/4, 441
distinct_combination_index/4, 444
distinct_combinations/3, 440
distinct_combinations/4, 441
distinct_derangement/2, 645
distinct_derangement/3, 646
distinct_derangement_index/3, 649
distinct_derangements/2, 645
distinct_derangements/3, 646
distinct_multiset/3, 2333
distinct_multiset/4, 2334
distinct_multiset_index/4, 2336
distinct_multisets/3, 2333
distinct_multisets/4, 2333
distinct_partition/2, 1625, 2688
distinct_partition/3, 1625, 2688
distinct_partition/4, 1626, 2689
distinct_partition_index/3, 1631, 2693
distinct_partition_index/4, 1631, 2694
distinct_partition_index/5, 1631, 2694
distinct_partitions/2, 1624, 2687
distinct_partitions/3, 1625, 2688
distinct_partitions/4, 1626, 2689
distinct_permutation/2, 2745
distinct_permutation/3, 2746
distinct_permutation_index/3, 2751
distinct_permutations/2, 2745
distinct_permutations/3, 2746
distinct_values/1, 1963
distribution/1, 51
dit_metric, 416
diversity/2, 976

djb2_32, 1136
djb2_64, 1137
doc_goal/1, 799
doc_metric, 417
doclet, 798
doctype/1, 1247
document/1, 2498, 3089
document/2, 2512, 3089
document/3, 1402
document/4, 1403
document_generation//2, 3828
dot//1, 1056
dot_graph_language, 697
dot_product/3, 873, 2093, 2934
double_metaphone/3, 3321
double_metaphone_match/2, 3322
dowhile/2, 2179
downcase_text/2, 3755
drop/3, 3624
dump_trace, 629
duration_between/3, 583
duration_string/2, 1689
during/2, 1671
dut1_entries/1, 3441, 3450
dut1_offset_at_utc_unix/3, 3442
dut1_override_data_/3, 3444
dut1_source/1, 3441, 3450
dut1_source_/1, 3445
dynamic_directive_/3, 3820

E

earliest/3, 3123
earliest_schedule/2, 3118
easter_day/3, 1692
ecb, 166
eclat_pattern_miner, 800
ed25519, 534
ed25519_keypair/2, 535
ed25519_public_key/2, 536
ed25519_sign/3, 536
ed25519_verify/3, 537
edb_fact_/1, 565
edcg, 801
edge/3, 1091
edge/4, 1101
edge/5, 673, 1763
edge/6, 714
edge_/5, 679
edge_case/2, 63
edge_counter_/1, 657
edge_index_/1, 729
edges/2, 1071
edit_similarity/3, 3317
edit_similarity/4, 3317
edmonds_karp, 919
edmonds_karp(Dictionary), 920
effective_maximum_pattern_length/3, 2732
effective_support_count/3, 2732
either, 822
either/3, 834
el_language_profile, 1882
elastic_net_regression, 804
elicit_counter_/1, 2211
elicit_request/5, 2206
elicit_url_request/5, 2213
eliminate_rows/5, 2989
elo_ranker, 805
email_hash/2, 1110
emeans_clusterer, 807
emeans_clusterer(Random), 808
emit_progress/5, 2233
emit_progress/6, 2215
empty/1, 113, 637, 746, 1070, 1189, 1675, 2186, 2440, 2575, 2790, 3103, 3548, 3607, 3664
empty_map/1, 3826
empty_search/2, 1722
en_language_profile, 1883
enable/0, 2140
enable/1, 608, 2140
enable/2, 608
enable_logging/1, 2175
enabled/1, 608
enabled/2, 609
enabled_/0, 2143
enabled_/1, 610
enabled_/2, 610
encode/2, 1799, 3205
encode/3, 987, 3205
encode_binary_data/2, 2322
encode_bits/3, 994
encode_body/4, 1287, 1300
encode_fixed_header/4, 2326
encode_frame/2, 22
encode_instance/3, 377, 785, 2974
encode_instance_checked/3, 2979
encode_int/3, 992
encode_linear_instance/3, 356
encode_message/4, 1556
encode_packet/2, 2315
encode_properties/2, 2323
encode_property/2, 2325
encode_property_list/2, 2324
encode_uint16/2, 2318
encode_uint32/2, 2319
encode_uint8/2, 2318
encode_utf8_string/2, 2321
encode_utf8_string_pair/2, 2323
encode_varint/2, 2320
encoded_feature_count/2, 2973
encoder_attribute_names/2, 786, 2978
encoder_block/3, 1958
encoding_suffix/2, 2287

encrypt/4, 166, 2700
encrypt/5, 160
encrypt_block/3, 158
encrypt_padded/4, 168
encrypt_padded/5, 162
encrypt_prepared_block/3, 155
endian_order/2, 316
endpoint/5, 1116, 2509, 3009
english_lemmatizer, 1969
english_tokenizer, 3507
ensure/2, 1480
ensure_absolute_url/2, 2549
ensure_directory/1, 2604
ensure_file/1, 2607
ensure_non_zero/1, 2988
ensure_secure_url/3, 2550
entailment_explanations/5, 1661
entails/4, 1659, 3122
entails/5, 1659, 3122
entity/1, 398, 599, 737, 1201, 2383
entity/2, 599, 737, 2383
entity_calls/3, 412
entity_defines_/2, 2762
entity_diagram, 698
entity_diagram(Format), 700
entity_file_/2, 2001
entity_info_pair_score_hook/3, 419
entity_info_score_hook/2, 419
entity_kind/2, 412
entity_mutants/2, 2386
entity_mutants/3, 2387
entity_predicate_breakpoint_/4, 616
entity_predicates_weights_hook/2, 419
entity_prefix/2, 493
entity_property/2, 412
entity_score/2, 402
entity_updates/3, 413
entry/1, 506
enumerate/2, 2809
environment_variable/2, 2607
eo_language_profile, 1884
epoch/1, 307
epoch/2, 313
epsg_crs/2, 511
epsilon/1, 2014
equal/2, 114, 1672, 3103
equality_constraints/2, 460
equality_jacobian/2, 460
equirectangular_inverse/4, 1009
equirectangular_projection/4, 1009
equivalent/2, 1666, 3753
erase/1, 2942
error/2, 1803
error_code/2, 1803
error_data/2, 1804
error_message/2, 1804
error_response/2, 3065
error_response/4, 1797
error_response/5, 1797
es256_signature_der/2, 1830, 2541
es_language_profile, 1885
essentially_equal/3, 2013, 3632
estimate_temperature/1, 3130
estimate_temperature/2, 3130
et_language_profile, 1886
eu_language_profile, 1887
euclidean_distance/3, 3639
euclidean_norm/2, 2093, 3638
euler_ode_solver(System), 2456
evaluate/2, 2486
evaluate/3, 1784, 1792, 2463
evaluate_derivative/3, 2466
evaluate_function/2, 2480
evaluate_integrand/2, 2473
evaluate_relative/4, 1793
event/2, 2506
event_interval/4, 3126
event_registry, 811
event_registryp, 812
event_relation/6, 3127
evict/3, 2188
ewma_anomaly_detector, 820
exactly_representable/1, 1618
example/2, 386, 794
example/3, 29, 364, 796, 797, 862, 2965
example_attribute_values/2, 779
example_count/1, 862
examples_to_linear_rows/3, 356
examples_to_rows/3, 376, 785, 2974
exception/4, 3827
exchange/3, 1497
exchange/4, 1277, 1499
exchange/5, 1499
exchange_bind/4, 10
exchange_code/5, 2529, 2537
exchange_declare/3, 9
exchange_delete/3, 9
exchange_sequence/3, 1498
exchange_sequence/4, 1278, 1500
exchange_unbind/4, 10
exclude/2, 3246
exclude/3, 2262
excluded/4, 1660
execution_context/7, 495
exists/3, 2952
expand/2, 996, 1771
expand/3, 3749
expand/4, 3750
expand_int/3, 996
expand_library_alias_paths, 821
expand_library_path/2, 490
expand_weights/3, 2990

expanding, 480
expected, 825
expected(Expected), 830
expected/1, 833
expecteds/2, 824
expire/4, 2953
explain//1, 3570
explain/2, 576
explicit_tracing_/0, 612
exponential/2, 2818, 3740
exponential_density/3, 3741
exponential_distribution/3, 3742
exponential_log_density/3, 3742
exponential_quantile/3, 3743
exponential_samples/3, 3741
exponential_smoothing, 840
exponential_smoothing_common, 843
exponential_smoothing_problem(Method,Series,Opts,Opts2,Opts3,Series,Frequency,ParameterSpecificatio
848
export/1, 3090
export/2, 3089
export_to_clauses/4, 39, 108, 361, 383, 791, 890, 2735, 2862, 2929, 2994, 3491
export_to_file/4, 39, 108, 362, 384, 792, 890, 2736, 2862, 2930, 2994, 3492
extend_catalog/3, 173, 1731, 3417
extension/1, 1220
extension_type/2, 2283
extension_type/3, 2283
extensions/1, 1220
external_predicate_/1, 738
external_reference/2, 54
extract_aos_packets/3, 256
extract_components/5, 784
extract_diagnostics/2, 1650
extract_packets/3, 225
extract_packets/4, 241
extract_scores/2, 33
extract_tc_segment/2, 289
extract_tm_packets/3, 253
extract_values/3, 2168

F

f/4, 3784
fa_language_profile, 1888
factorial/2, 3626
facts/1, 577
fail_insertion(Entity,Predicate,ClauseIndex,Occurrence,Mutation),
2377
failed_/3, 2042
failed_count_/1, 2409
failed_test_reason//1, 2044
failure_details/3, 3160
fallback_rating/5, 2916
falling_factorial/3, 3627
false/1, 1749
fast_random, 2796
fast_random(Algorithm), 2797
fault/5, 3827
fcube, 849
fcube/0, 850
feature_dataset_protocol, 861
feature_discretization, 863
feature_names/2, 880
feature_pair_mutual_information/3, 866
feature_penalty_profiles/5, 2984
feature_redundancy, 865
feature_scores/2, 888
feature_scoring_common, 866
feature_scoring_protocol, 875
feature_selector_common, 876
feature_selector_protocol, 884
feature_values/3, 881
fetch/2, 221, 224
fi_language_profile, 1889
fibonacci/2, 3626
field/3, 1110
field/4, 1379
field_part/4, 1380
fields/2, 1376, 2448
fields/5, 1612
file/1, 398, 600, 701, 1990, 3816
file/2, 398, 600, 701, 1990, 3816
file/6, 1379
file_being_advised_/4, 3822
file_dependency_diagram, 703
file_dependency_diagram(Format), 704
file_diagram(Format), 706
file_exists/1, 2604
file_footer/3, 713
file_header/3, 712
file_line_hit_count_/3, 618
file_load_diagram, 709
file_load_diagram(Format), 710
file_modification_time/2, 2605
file_part/6, 1380
file_path/2, 2017
file_path_components/2, 3754
file_permission/2, 2605
file_score/2, 403
file_size/2, 2605
file_to_bytes/2, 2893
file_to_bytes/3, 2893
file_to_chars/2, 2891
file_to_chars/3, 2892
file_to_codes/2, 2891
file_to_codes/3, 2891
file_to_terms/2, 2892
file_to_terms/3, 2892
file_type_extension/2, 492
file_url/2, 2018
files/1, 667, 688, 1990, 3815

files/2, 667, 687, 1377, 1989, 3815
files/3, 667, 687
filter/2, 2580
filter/3, 836, 3774
filter_external_file_extension/3, 676
filter_feature_scores/6, 893
filter_feature_selector_common, 891
filter_file_extension/3, 676
filter_model/1, 892
filter_patterns/3, 962, 3096
filter_scoring_metric/2, 892
filter_selection/3, 894
filter_valid_preparation_diagnostics/2, 894
filter_validate_dataset/3, 893
filter_validate_diagnostics/2, 894
final/2, 1547
final_bearing/3, 1012
final_hash_state/2, 1158
find/4, 3710
find/5, 3710
find_root/2, 2482
find_root/3, 2482
find_root/4, 2483
findall_in_noblock/3, 2077
findall_in_noblock/4, 2077
findall_member/4, 2262
findall_member/5, 2262
findall_rd_noblock/3, 2077
findall_rd_noblock/4, 2076
fine/2, 303
finish/3, 1479
finished_by/2, 1672
finishes/2, 1672
finite_binary_rational/4, 1613
finite_number/1, 844
fips202_fixed_hash(RateBytes,Suffix,OutputBytes), 1138
fips202_hash(RateBytes,Suffix,OutputBytes), 1139
fired_/3, 2043
first_header_pointer/2, 234
first_significant_component/2, 2116
first_significant_component/3, 2116
fisher/3, 2816, 3743
fisher_density/4, 3745
fisher_distribution/4, 3746
fisher_log_density/4, 3745
fisher_quantile/4, 3747
fisher_samples/4, 3744
fisher_score, 895
fisher_score_feature_selector, 905
fit/2, 2462
fit/3, 2462
fit_level/13, 3432
fit_linear_model/7, 2975
fit_paired_series/4, 1644
fit_ridge_model/7, 2975
fit_series/4, 1646
fit_series_output/5, 1646
fit_smoothing/9, 844
fitness/2, 774, 2681
fitted_model/3, 1647
fitted_values/2, 1639, 3426
fitted_values/3, 1640, 3427
fix_node2_left_underflow/6, 759
fix_node2_right_underflow/6, 759
fix_node3_left_underflow/9, 760
fix_node3_middle_underflow/9, 760
fix_node3_right_underflow/9, 760
fix_option/2, 2591
fix_options/2, 2591
flag_group_chk/1, 909
flag_groups/1, 909
flag_value_/2, 912
flags, 906
flags_validator, 914
flaky_/1, 2042
flat_map/2, 834, 2581, 3775
flatten/1, 839, 2584, 3780
flatten/2, 1772, 3608, 3664
flatten_goals//1, 3796
flatting, 3795
float, 3597
float//1, 1055
flow_balances/3, 928
flow_cost/3, 927, 935
flow_list/3, 935
flow_network, 921
flow_network(Dictionary), 923
flow_network_protocol, 924
flow_support(Dictionary), 931
flush_all/1, 2254
flush_all/2, 2255
fnv1a_32, 1141
fnv1a_64, 1142
fold_diacritics/2, 3390
fold_diacritics_codes/3, 3382
fold_left/4, 2263
fold_left_1/3, 2264
fold_right/4, 2265
fold_right_1/3, 2265
footer/2, 2726
forbidden_response/3, 1391
forbidden_response/4, 1391
fordownto/3, 2181
fordownto/4, 2182
fordownto/5, 2182
foreach/3, 2180
foreach/4, 2180
forecast/3, 3489
forecast_error_metrics/3, 3468
forecast_interval/5, 841, 3497

forecast_smoothing/4, 845
forecaster_common, 3456
forecaster_diagnostics_data/2, 3459
forecaster_export_template/4, 3459
forecaster_options/2, 3491
forecaster_protocol, 3487
forecaster_term_template/2, 3459
form_body/2, 3005
form_data_body/2, 1381
format, 958
format/1, 306
format/2, 309, 313, 959
format/3, 959
format_content_items/2, 2222
format_date_time/4, 584
format_entity_score//2, 404
format_object/1, 668
format_prompt_messages/2, 2223
format_report/1, 2386
format_report/2, 2386
format_report/3, 2385
format_resource_contents/2, 2223
format_to_atom/3, 3375
format_to_chars/3, 3375
format_to_chars/4, 3375
format_to_codes/3, 3376
format_to_codes/4, 3376
format_token/4, 2715
formatted_iban/2, 1695
forto/3, 2180
forto/4, 2181
forto/5, 2181
forward/1, 482
forward/2, 3839
forward/3, 3840
forward_substitution/3, 2118
forwarding, 481
fp_growth_pattern_miner, 960
fr_language_profile, 1890
fractile/3, 3224
frame/2, 2506
frame/3, 1772
frame/5, 1543
frame_body/2, 3242
frame_command/2, 3241
frame_header/3, 3241
frame_headers/2, 3241
frame_type/2, 219, 222
freeze/2, 500
frequency/1, 3495
frequency_distribution/2, 3228
frequent_pattern_mining_common, 961
frobenius_norm/2, 2099
from_expected/2, 3770
from_generator/2, 829, 2576, 3769
from_generator/3, 828, 2576, 3769
from_generator/4, 828, 3769
from_goal/2, 828, 2576, 3768
from_goal/3, 827, 2575, 3768
from_goal/4, 827, 3768
from_goal_or_throw/2, 2577
from_goal_or_throw/3, 2577
from_graph/2, 926
from_list/2, 114
from_optional/3, 829, 3770
from_unix_seconds/2, 307
from_uuid/3, 3575
frozen/2, 501
full_device_path/1, 2602
func_test/3, 2275
functional/0, 2276
fund_currency/5, 1704, 1705

G

ga_language_profile, 1891
game/4, 2889
gamma/3, 2819, 3732
gamma_density/4, 3733
gamma_distribution/4, 3735
gamma_log_density/4, 3734
gamma_quantile/4, 3735
gamma_samples/4, 3733
gat/4, 2253
gats/5, 2254
gauss_legendre_quadrature(Function), 2458
gaussian_mixture_clusterer, 966
gaussian_process_regression, 967
gc/2, 1483
generalized_ripple_down_rules, 3013
generate/1, 561, 1603, 1873, 2434, 3172, 3574, 3701
generate/2, 131, 133, 135, 137, 139, 141, 192, 212, 217, 277, 305, 505, 1001, 1039, 1245, 1545, 1617, 1760, 1765, 1770, 1777, 1784, 1790, 2258, 2513, 3067, 3525, 3537, 3574, 3701, 3752, 3809, 3832
generate/3, 129, 217, 278, 306, 312, 505, 1375, 1617, 2785, 3575
generate/4, 129, 2786, 3068
generate/8, 3702
generate_all/2, 3832
generate_authentication_info/2, 1349
generate_authorization/2, 1252, 1348, 1385
generate_body/3, 1298
generate_challenge/2, 1251, 1346, 1384
generate_cookie/2, 1280
generate_elicit_id/1, 2210
generate_fragment/2, 1791
generate_frame/3, 238
generate_frames/3, 239
generate_headers/2, 1297
generate_idle_packet/4, 274
generate_masking_key/1, 1596
generate_relative/2, 1792

generate_request/2, 1293
generate_request_line/2, 1295
generate_response/2, 1294
generate_response_headers/2, 1294
generate_scope_value/2, 1395
generate_set_cookie/4, 1281
generate_status_line/2, 1296
generated_predicate_/1, 2270
generating_/0, 2053
genetic_algorithm(Problem), 969
genetic_algorithm(Problem,Random), 970
genetic_algorithm_problem_protocol, 973
genint, 978
genint/2, 980
genint_core, 979
gensym, 981
gensym/2, 983
gensym_core, 982
geohash, 984
geohash_protocol, 986
geohash_to_int/2, 993
geojson, 998
geojson(ObjectRepresentation,PairRepresentation,StringRepresentation), 999
geojson_protocol, 1000
geojson_to_json/2, 1004
geometric/2, 2817
geometric_mean/2, 3219
geospatial, 1005
geospatial_protocol, 1006
get/1, 2582
get/3, 1264, 1481, 2250, 2947, 3014
get/4, 1264, 1340, 1462, 2187, 2250
get_field/2, 1744
get_flag_value/2, 908
get_object/5, 3037
get_object/6, 3047
get_seed/1, 63, 2801
getrange/5, 2948
gets/4, 2251
gets/5, 2251
git, 1028
git_object_identifier/1, 56
git_protocol, 1029
gl_language_profile, 1892
glicko2_common, 2836
glicko2_periodic_ranker, 1034
glicko2_ranker, 1036
global_mean_rating/2, 2918
gnu/0, 850
goal_expansion/2, 481
gomory_hu, 936
gomory_hu(Dictionary), 937
gpx, 1037
gpx_protocol, 1038
gpx_to_xml/2, 1041
gradient/2, 2156
gradient_boosting_classifier, 1042
gradient_boosting_regression, 1044
gradient_descent(Problem), 2150
gradient_matrix/6, 3557
gram_matrix/2, 2096
grammar_rules_hook, 1231
graph/2, 1763
graph_coloring/3, 1082
graph_common, 1066
graph_connected/2, 3706
graph_footer/5, 713
graph_header/5, 713
graph_language_protocol, 711
graph_language_registry, 715
graph_protocol, 1068
graph_types, 1079
gravatar, 1107
gravatar_api, 1115
ground/1, 3651
ground_entity_identifier/3, 675
ground_track/5, 3505
gtin_string/6, 3506
group/1, 2881
group_by_key/2, 3648
group_by_target/2, 874
group_consecutive_by_key/2, 3648
group_scores/3, 1958
group_sorted_by_key/2, 3647
grouped_dataset_groups/2, 2873
grouped_dataset_item_relevance_scores/3, 2879
grouped_dataset_item_relevances/4, 2878
grouped_dataset_items/2, 2874
grouped_dataset_items/3, 2874
grouped_dataset_relevance_frequencies/2, 2878
grouped_dataset_summary/2, 2874
grouped_dataset_tie_blocks/4, 2879
grouped_strength_ranker_common, 2838
gsp_pattern_miner, 1117
guess_all_extensions/2, 2284
guess_all_extensions/3, 2285
guess_arity/2, 557
guess_extension/2, 2284
guess_extension/3, 2284
guess_file_type/3, 2282
guess_file_type/4, 2283
guess_separator/2, 557
guess_type/3, 2282
guess_type/4, 2282
gumbel/3, 2822

H

ha1/4, 1355
ha_language_profile, 1893

halstead_metric, 421
halstead_metric(Stroud), 422
halt/0, 492
halt/1, 492
halt_hook/2, 493
hamming/3, 3316
hamming_distance/3, 3608
handbook/0, 1200
handle/2, 1309, 1427, 1525, 1576
handle_completion_complete/4, 2209
handle_initialize/4, 2207
handle_mcp_request/4, 2232
handle_message/3, 2226
handle_open_ai/3, 2501
handle_ping/2, 2207
handle_prompts_get/4, 2208
handle_prompts_list/3, 2208
handle_resources_list/3, 2209
handle_resources_read/4, 2210
handle_resources_templates_list/3, 2209
handle_root_underflow/2, 758
handle_seed_/1, 1522, 1541
handle_state_/9, 1522, 1541
handle_tools_call/4, 2208
handle_tools_list/3, 2207
harmonic_mean/2, 3220
has_aggregate_rules/0, 569
has_cycle/1, 1064, 1082
has_negative_cycle/1, 1103
has_pair/3, 2223
has_path/3, 1075
hash/2, 1111, 1156
hash_common_32, 1143
hash_common_64, 1148
hash_digest_protocol, 1153
hash_protocol, 1155
hash_state_protocol, 1157
haversine_distance/3, 1009
hdbscan_clusterer, 1181
hdel/4, 2955
he_language_profile, 1894
head/2, 2790
head/3, 1265
head/4, 1266, 1340, 1462
head_arguments_mutation(Entity,Predicate,Clause,Index,Occurrence,PrintMutation), 2378
head_arguments_reordering(Entity,Predicate,Clause,Index,Occurrence,PrintMutation), 2379
head_bucket/3, 3033
head_bucket/4, 3050
head_object/4, 3036
head_object/5, 3051
head_pred/1, 2274
header/2, 1323, 1326, 1813, 1826
header/3, 1302
header_algorithm/2, 1836, 2554
header_key_id/2, 1836, 2554
header_value/3, 3031
headers/1, 1323, 1326
headers/2, 1302
heap(Order), 1186
heap_protocol, 1187
heapp, 1192
help, 1198
help/0, 1200, 2652
help/1, 450
help/2, 454
help/3, 455
hessian/2, 2156
heuristic/3, 45
heuristic_expansion(Mode), 3796
hex_bytes/2, 523
hex_digest/4, 393, 1223
hex_digest/5, 394, 1224
hex_digit//1, 1054
hex_digits//1, 1054
hexists/4, 2955
hget/4, 2954
hgetall/3, 2955
hi_language_profile, 1895
hierarchical_clustering, 1205
hierarchyp, 1215
high_surrogate/1, 316
hkdf/5, 525
hkeys/3, 2956
hlen/3, 2956
hmac, 1221
hmac_hash/2, 1837
hmac_protocol, 1222
hodge_rank, 1225
home/1, 2617, 2674
homepage/1, 50
hook_pipeline(Pipeline), 1226
hook_set(Set), 1228
host_labels/2, 1320
hotp/5, 2613
hotp_verify/7, 2614
hr_language_profile, 1896
hset/5, 2954
html, 1244
html5, 1247
http_authenticate, 1249
http_authentication_error_printer, 1255
http_authentication_helpers, 1284
http_body_codec_protocol, 1286
http_client, 1261
http_client_core, 1275
http_client_digest_session, 1335
http_client_digest_session(HTTPTransport), 1336
http_client_session, 1457
http_client_session(HTTPTransport), 1458

http_cookie_jar, 1466
http_cookies(Representation), 1279
http_core, 1289
http_cors, 1331
http_digest, 1344
http_digest_verifier_protocol, 1354
http_directory_listing, 1359
http_docroot_paths, 1304
http_form_body_codec, 1307
http_handler_protocol, 1308
http_htmx, 1362
http_htpasswd_verifier(Path), 1257
http_json_body_codec, 1309
http_json_term_helpers, 1310
http_message_helpers, 1314
http_multipart, 1373
http_oauth, 1381
http_oauth_exact_scope_checker, 1393
http_oauth_helpers, 1394
http_oauth_introspection_client, 1397
http_oauth_introspection_verifier(Endpoint,Options), 1399
http_oauth_jwt_verifier(KeyOrJWKSet,Options), 1400
http_oauth_metadata, 1401
http_oauth_scope_checker_protocol, 1405
http_oauth_verifier_protocol, 1406
http_octet_stream_body_codec, 1317
http_origin_site_helpers, 1318
http_parameters, 1412
http_process_transport, 1418
http_request_protocol, 1321
http_response_protocol, 1324
http_router, 1426
http_router_basic_auth(Verifier,BaseOptions), 1258
http_router_digest_auth(Verifier,ProtectOptions,AuthenticationInfoOptions), 1355
http_router_htmx, 1371
http_router_oauth(Verifier,BaseOptions), 1407
http_router_parameters, 1416
http_router_server_session(Manager), 1474
http_server, 1436
http_server_core, 1451
http_server_core_basic_handler(Verifier,Handler,ProtectOptions), 1259
http_server_core_digest_handler(Verifier,Handler,ProtectOptions,AuthenticationInfoOptions), 1358
http_server_core_oauth_endpoint_handler(ProtectedResource,MetadataDescriptors,MetadataOptions), 1409
http_server_core_oauth_handler(Verifier,Handler,ProtectOptions), 1411
http_server_core_session, 1476
http_server_core_session_event/2, 1484
http_server_core_session_handler(Manager,Handler), 1486
http_socket_transport, 1487
http_sse, 1507
http_sse_accept_handler(Options), 1523
http_sse_service_handler_protocol, 1524
http_static_files, 1526
http_text_body_codec, 1327
http_text_helpers, 1328
http_transport_protocol, 1491
http_websocket, 1528
http_websocket_accept_handler(Options), 1541
http_websocket_client_service, 1557
http_websocket_client_service(HTTPTransport), 1558
http_websocket_client_session, 1585
http_websocket_frames, 1542
http_websocket_handshake, 1549
http_websocket_messages, 1551
http_websocket_messages(TextRepresentation), 1552
http_websocket_server_service, 1562
http_websocket_server_service(HTTPTransport), 1563
http_websocket_server_session, 1586
http_websocket_service(HTTPTransport,Role,TextRepre
1570
http_websocket_service_handler_protocol, 1575
http_websocket_service_loop_options, 1577
http_websocket_service_registry, 1578
http_websocket_session(Role,TextRepresentation), 1587
http_websocket_session_read_options, 1597
http_websocket_session_write_options, 1598
https_url/1, 2549
human_language_profile, 1887
hvals/3, 2956
hy_language_profile, 1898
hybrid_language_detector, 1899
hyperedge/4, 1763
hyperedge/5, 1764
hypergeometric/4, 2817

I

iban/4, 1695
ibk/3, 2275
ica_projection, 1600
id/2, 1802
id_language_profile, 1900
idb_fact_/1, 565
iddfs_interpreter(Increment), 3797
identity_hook, 1232
identity_matrix/2, 2095
ids, 1601
ids(Representation,Bytes), 1602

ieee_754(Precision,ByteOrder,NaNRepresentation), 1604
ieee_754_common(Precision,ByteOrder), 1605
ieee_754_fields(Precision,ByteOrder), 1610
ieee_754_fields_protocol, 1611
ieee_754_protocol, 1615
if_empty/1, 2579
if_expected/1, 831
if_expected_or_else/2, 832
if_invalid/1, 3773
if_present/1, 2580
if_present_or_else/2, 2580
if_unexpected/1, 832
if_valid/1, 3772
if_valid_or_else/2, 3773
in/1, 2073
in/2, 2072
in_degree/3, 1063
in_label/3, 124
in_list/2, 2074
in_list/3, 2074
in_noblock/1, 2073
in_noblock/2, 2073
in_nodes_section_/0, 656
in_use/2, 27
include/3, 2261
included_caller_/1, 655
included_directory_/1, 694
included_entity_/1, 702
included_file_/1, 708
included_library_/2, 724
included_module_/1, 702
included_predicate_/1, 737
incr/3, 2950
incr/4, 2252
incrby/4, 2951
increase/1, 3788
increment/0, 3788
increment_counter/1, 2063
index_items/3, 2834, 2849, 2853
indexed_known_values/5, 3476
indexed_series_observations/3, 3463
inequality_constraints/2, 460
inequality_jacobian/2, 460
inheritance_diagram, 716
inheritance_diagram(Format), 717
init/0, 3806
init/2, 3337
init1/2, 3338
init_log_file/2, 2174
init_tail/2, 3340
init_tails/2, 3340
initial_bearing/3, 1012
initial_channel_reassembly_state/1, 250
initial_parameter_point/2, 846
initial_point/1, 2156
initial_point/2, 2160
initial_population/1, 974
initial_positions/1, 775, 2681
initial_reassembly_state/1, 240, 249, 288
initial_state/1, 268, 1588, 3133, 3359
initial_strengths/2, 2839, 2849
initial_temperature/1, 3134
initial_vectors/2, 784
initial_velocities/1, 2681
initialize/1, 24
initialize_scores/2, 2886
initialized_/0, 2212
inits/2, 3336
inits1/2, 3338
inner_progress/6, 461
inorder/2, 741
input_schema/2, 2242
insert/3, 3104
insert/4, 745, 1188, 3549
insert_after/3, 3841
insert_all/3, 1188, 3104
insert_aos_packets/4, 256
insert_before/3, 3840
insert_empty/4, 762
insert_impl/4, 761
insert_in/4, 2442
insert_node2_2/4, 762
insert_node2_4/4, 762
insert_node3_4/4, 763
insert_node3_7/4, 763
insert_packets/4, 226
insert_packets/5, 241
insert_tc_segment/3, 289
insert_tm_packets/4, 253
insert_zone/2, 214
install/1, 2636
install/2, 2636
install/3, 2635
install/4, 2634
installed/0, 2623
installed/1, 2623
installed/3, 2623
installed/4, 2622
instance/1, 1210
instance/2, 2943
instances/1, 1210
instant_to_utc_date_time/2, 3453
int_to_geohash/3, 994
integer, 3599
integer//1, 1055
integer_partitions, 1620
integer_partitions_protocol, 1621
integer_to_bytes/4, 194
integer_to_bytes/5, 194
integrate/3, 2475
integrate/4, 2475

integrate/5, 2476
integrate_series/3, 3467
intermittent_demand_forecasting, 1637
internal_apostrophe/1, 3520
internal_error/2, 1799
internal_hyphen/1, 3520
internal_os_path/2, 2599
interpolate_great_circle/4, 1013
interpolate_rhumb/4, 1011
interpolator, 2459
interpolator_protocol, 2461
interpretations/2, 125
interpreterp, 3798
interquartile_range/2, 3225
intersect/2, 3104
intersection/2, 748
intersection/3, 748, 1677, 3105
intersection/4, 3105
intersection_impl/4, 763
interval, 1652
interval_algebra, 1653
interval_algebra_protocol, 1654
interval_constraint_network, 1656
interval_constraint_network_protocol, 1657
interval_protocol, 1666
interval_relation_set, 1673
interval_relation_set_protocol, 1674
interval_string/2, 1690
introspect/4, 1398
invalid/1, 3774
invalid_params/2, 1799
invalid_request/1, 1798
invalids/2, 3763
inverse_matrix/2, 2108
invert_from_cholesky/2, 2120
invocation_number_/1, 614
invoke/1, 1745
invoke/2, 1745
ip_grammars(Format), 1050
ipv4//1, 1051
ipv6//1, 1051
iqr_anomaly_detector, 1679
is_absolute_file_name/1, 2597
is_acyclic/1, 1064
is_alpha/1, 3584
is_alphanumeric/1, 3584
is_ascii/1, 3584
is_batch/1, 1801
is_bin_digit/1, 3585
is_bipartite/1, 1078
is_body/1, 1374
is_boosted_request/1, 1364
is_clockwise_polygon/1, 1018
is_complete/1, 1078
is_connected/1, 1081
is_control/1, 3589
is_cors_request/1, 1332
is_dec_digit/1, 3586
is_empty/0, 2579
is_end_of_line/1, 3590
is_error_response/1, 1801
is_expected/0, 831
is_false/1, 1750
is_fragment_request/1, 1364
is_frame/1, 1544
is_hex_digit/1, 3586
is_history_restore_request/1, 1364
is_htmx_request/1, 1363
is_invalid/0, 3772
is_layout/1, 3588
is_letter/1, 3585
is_lower_case/1, 3586
is_message/1, 1554
is_newline/1, 3589
is_node4/1, 764
is_notification/1, 1800
is_null/1, 1751
is_object/1, 1751
is_octal_digit/1, 3585
is_part/1, 1378
is_period/1, 3589
is_prefix_of/2, 3345
is_preflight_request/1, 1332
is_present/0, 2579
is_punctuation/1, 3588
is_quote/1, 3588
is_request/1, 1291, 1800
is_response/1, 1292, 1801
is_sparse/1, 1078
is_state/1, 1589
is_stop_word/1, 3246
is_subsequence_of/2, 3342
is_suffix_of/2, 3345
is_tree/1, 1081
is_true/1, 1750
is_unexpected/0, 831
is_upper_case/1, 3587
is_valid/0, 3772
is_valid_polygon/1, 1019
is_validator/1, 913
is_void/1, 1750
is_vowel/1, 3587
is_white_space/1, 3587
iso8601, 1680
iso_13616, 1693
iso_13616_protocol, 1694
iso_13616_registry, 1696
iso_3166, 1697
iso_3166_1, 1698
iso_3166_2, 1700
iso_3166_protocol, 1701
iso_4217, 1703

iso_4217_protocol, 1704
iso_639, 1706
iso_639_1, 1707
iso_639_2, 1708
iso_639_3, 1710
iso_639_5, 1711
iso_639_protocol, 1712
iso_8859_1, 320
iso_8859_10, 321
iso_8859_13, 322
iso_8859_14, 323
iso_8859_15, 324
iso_8859_16, 325
iso_8859_2, 326
iso_8859_3, 327
iso_8859_4, 328
iso_8859_9, 329
iso_9362, 1715
iso_9362_protocol, 1716
isolation_forest_anomaly_detector, 1718
issue_creator, 1720
it_language_profile, 1902
item/1, 2843, 2845, 2847, 2889, 2901
item/2, 2881
item_content/2, 2901
item_content_dataset_protocol, 2900
item_content_dataset_validation, 2901
item_denominator/5, 2852
item_mean_rating/3, 2919
item_vector/3, 2918
items/1, 965, 3093
items/2, 2917
iterative_dimension_reducer_diagnostics/11, 782
iterator_element/2, 1753

J

ja_language_profile, 1903
jaccard_index/3, 3319
jaccard_recommender, 1729
jaccard_similarity, 2905
jaccard_similarity/3, 2920
jar_cookie_/11, 1473
jar_seed_/1, 1473
jar_state_/2, 1473
jaro/3, 3316
jaro_winkler/3, 3317
java, 1740
java(Reference), 1741
java(Reference,ReturnValue), 1742
java_access_protocol, 1743
java_hook, 1746
java_utils_protocol, 1747
jira_jql/3, 1723
jira_label_conditions/2, 1724
jira_list_command/4, 1723
jira_search_command/3, 1723
job_title/2, 1113
join/3, 2791
join_all/3, 2791
join_aos_packet_zone/3, 254
join_packet_zone/4, 253
json, 1755
json(ObjectRepresentation,PairRepresentation,String 1758
json(StringRepresentation), 1757
json_array_body/2, 3004
json_body/2, 3003
json_bytes/2, 2712
json_graph, 1761
json_graph_data_protocol, 1762
json_graph_protocol, 1764
json_ld, 1766
json_ld(ObjectRepresentation,PairRepresentation,Str 1768
json_ld(StringRepresentation), 1767
json_ld_protocol, 1769
json_lines, 1773
json_lines(ObjectRepresentation,PairRepresentation, 1775
json_lines(StringRepresentation), 1774
json_lines_protocol, 1776
json_media_descriptor/2, 2517
json_media_descriptor/3, 2517
json_member/3, 1832, 2546, 2712
json_member_default/4, 1832, 2546
json_object/1, 1832, 2712
json_object_body/2, 3004
json_object_pairs/2, 1311, 1832, 2552, 2712
json_patch, 1778
json_patch_protocol, 1779
json_path, 1781
json_path(StringRepresentation), 1782
json_path_protocol, 1783
json_pointer, 1787
json_pointer(StringRepresentation), 1788
json_pointer_protocol, 1789
json_protocol, 1759
json_request_body_descriptor/4, 2518
json_request_body_descriptor/5, 2519
json_response/4, 3006
json_response/5, 3007
json_response_descriptor/4, 2519
json_response_descriptor/5, 2520
json_rpc, 1794
json_schema, 1806
json_schema(ObjectRepresentation,PairRepresentation 1808
json_schema(StringRepresentation), 1807
json_schema_protocol, 1809
json_to_geojson/2, 1003
jump/3, 2791

jump_all/3, 2792
jump_all_block/3, 2792
jump_to_invocation_number_/1, 614
justifications/2, 124
justify/5, 119
jwks/3, 2531, 2537
jwt, 1811
jwt_claims, 1821
jwt_compact, 1824
jwt_der, 1829
jwt_helpers, 1831
jwt_jwa, 1835
jwt_jwk, 1838
jwt_jwks, 1840
jwt_jws, 1843
jwt_openssl, 1847

K

k_distinct_subsequence/3, 3349
k_distinct_subsequences/3, 3348
k_permutation/3, 2747
k_permutation/4, 2747
k_permutations/3, 2746
k_permutations/4, 2747
kcenters_clusterer, 1848
kemeny_young_ranker, 1849
kernel_pca_projection, 1852
kernel_svm_classifier, 1853
key/2, 3646
key_size/1, 159
key_type/2, 1837
keys/2, 750, 3646
keys/3, 2952
keys_values/3, 3646
keysort/2, 3608
kill/1, 2783
kill/2, 2782
kmeans_clusterer, 1854
kmedians_clusterer, 1856
kmedoids_clusterer, 1857
kmodes_clusterer, 1858
knn_classifier, 1859
knn_distance_anomaly_detector, 1861
knn_forecasting, 1863
knn_item_recommender, 1864
knn_regression, 1866
knn_user_recommender, 1867
known_attribute_values/3, 376, 785, 2976
known_zone_id/1, 3697
ko_language_profile, 1904
kprototypes_clusterer, 1868
ksuid, 1870
ksuid(Representation,Alphabet), 1871
ksuid_protocol, 1872
ku_language_profile, 1905
kurtosis/2, 3223

L

la_language_profile, 1906
label/3, 124
label_arguments/2, 1728
lagged_rows/3, 3467
lambert_azimuthal_equal_area/3, 512
lambert_azimuthal_equal_area_inverse/3, 512
language/1, 1924
language/3, 1708, 1713
language/5, 1711, 1714
language_code/5, 1709, 1713
language_detection_scoring, 1907
language_detection_scripts, 1909
language_detection_strategy_protocol, 1911
language_detection_tokenizer, 1913
language_detector(Representation,Strategy), 1914
language_detector_protocol, 1915
language_group/2, 1712, 1714
language_object/2, 716
language_profile_common, 1920
language_profile_expansion, 1922
language_profile_protocol, 1923
language_profiles, 1925
language_scripts/2, 1910
lasso_feature_selector, 1955
lasso_regression, 1965
last/2, 3609, 3664
last_time_sequence_/2, 3168
latest/3, 3123
lbfgs(Problem), 2152
lbfgs_b(Problem), 2153
lcom_metric, 424
lcov_report, 2000
lda_classifier, 1966
lda_projection, 1968
leading_magnitude/2, 2988
leaf/1, 1216
leaf_class/1, 1212
leaf_classes/1, 1213
leaf_instance/1, 1212
leaf_instances/1, 1212
leap_effective_date/2, 3442
leap_offset_at_utc_unix/2, 3442
leap_override_data_/2, 3444
leap_second_date/2, 3447
leap_seconds_entries/1, 3439, 3448
leap_seconds_source/1, 3439, 3448
leap_source_/1, 3445
leap_year/1, 581, 1691
leaping_/1, 613
learn/0, 2280
learn/1, 2280
learn/2, 35, 359, 381, 789, 886, 2272, 2859, 2925, 2992, 3406, 3489

learn/3, 35, 359, 381, 789, 885, 2272, 2859, 2923, 2992, 3406, 3488, 3555
learn_pair/4, 1644
learn_seq/2, 2273
learn_state/6, 1647
learn_transform/3, 3409
learn_transform/4, 3409
learn_with_timeout/4, 2273
leash/1, 623
leashing/1, 623
leashing_/1, 613
least_common_multiple/2, 3643
least_squares/3, 2109
leaves/1, 1216
leaves/2, 1087
lemma/2, 1972, 1974
lemma/3, 1972, 1974
lemmas/2, 1975
lemmas/3, 1976
lemmatizer(Representation,Language), 1970
lemmatizer_language_protocol, 1971
lemmatizer_protocol, 1973
length/2, 639, 2793, 3609, 3665
levenshtein/3, 3315
lgtdoc, 1978
lgtdoc_messages, 1982
lgtdocp, 1983
lgtunit, 2002
lgtunit_messages, 2044
libraries/0, 1201
libraries/1, 662, 682, 1986
libraries/2, 662, 682, 1986
libraries/3, 662, 682
library/1, 400, 602, 664, 684, 1201, 1987, 2382
library/2, 400, 602, 664, 684, 1986, 2382
library_dependency_diagram, 718
library_dependency_diagram(Format), 719
library_diagram(Format), 721
library_entity_/4, 1979
library_load_diagram, 725
library_load_diagram(Format), 726
library_mutants/2, 2388
library_mutants/3, 2388
library_score/2, 402
license/1, 50, 2617
linda, 2067
linda/0, 2081
linda/1, 2080
linda_client, 2069
linda_client/1, 2070
linda_client/2, 2070
linda_server, 2079
linda_timeout/2, 2072
line_to_bytes/2, 2897
line_to_bytes/3, 2898
line_to_chars/2, 2896
line_to_chars/3, 2896
line_to_codes/2, 2897
line_to_codes/3, 2897
linear_algebra, 2087
linear_algebra_protocol, 2088
linear_encoders_feature_count/2, 356
linear_programming_common, 2121
linear_programming_protocol, 2124
linear_programming_types, 2133
linear_regression, 2136
linear_regression/4, 3644
linear_svm_classifier, 2137
linear_trend_forecast/4, 3471
lines_covered_/1, 1995
lines_metric, 425
lines_total_/1, 1995
lint/0, 2647, 2671
lint/1, 2647, 2671
lint/2, 2647
linter_reporter, 2139
linter_warning_sequence_/1, 2041
list, 3602
list(Type), 3603
list/0, 2663
list_buckets/2, 3033
list_buckets/3, 3049
list_buckets_response/2, 3063
list_objects_v2/4, 3034
list_objects_v2/5, 3051
list_objects_v2_response/2, 3064
list_to_array/2, 1753
list_to_curly_pairs/2, 1312
listener_active_worker_/3, 1425, 1490
listener_endpoint_/3, 1491
listener_event_/2, 1423
listener_handle_alias_/2, 1423
listener_seed_/1, 1422
listener_shutdown_control_/3, 1424, 1490
listener_shutdown_requested_/2, 1425, 1490
listener_shutdown_seed_/1, 1424, 1489
listener_temporary_tls_credentials_/3, 1423
listing, 2144
listing/0, 2145
listing/1, 2145
listp, 3604
literal/2, 3086
literal_bucket/2, 571
llen/3, 2958
load/1, 2286, 3675
load/2, 1472, 2286, 3675
load_config/1, 2408
load_coverage_config/1, 2406
load_dut1_override/1, 3440, 3449
load_leap_seconds_override/1, 3439, 3448
load_program/1, 573
load_registry/1, 2658

loaded_file/1, 490
loaded_file_property/2, 490, 730
loaded_files_topological_sort/1, 491
loaded_files_topological_sort/2, 491
loaded_pack/3, 2631
loaded_pack_dependency/6, 2633
loaded_pack_file/4, 2631
loader_file/1, 51
local_abbreviation/2, 3690
local_abbreviation/3, 3689
local_abbreviation/4, 3689
local_abbreviation_reified/2, 3695
local_abbreviation_reified/3, 3695
local_abbreviation_reified/4, 3694
local_abbreviation_with_resolution/3, 3691
local_abbreviation_with_resolution/4, 3690
local_abbreviation_with_resolution/5, 3690
local_daylight_saving_time/2, 3687
local_daylight_saving_time/3, 3687
local_daylight_saving_time/4, 3687
local_daylight_saving_time_reified/2, 3694
local_daylight_saving_time_reified/3, 3694
local_daylight_saving_time_reified/4, 3693
local_daylight_saving_time_with_resolution/3, 3689
local_daylight_saving_time_with_resolution/4, 3688
local_daylight_saving_time_with_resolution/5, 3688
local_decrypt/3, 2721
local_decrypt/5, 2721
local_encrypt/3, 2719
local_encrypt/5, 2720
local_encrypt_with_nonce/6, 2728
local_key/1, 2719
local_offset/2, 3685
local_offset/3, 3685
local_offset/4, 3685
local_offset_reified/2, 3693
local_offset_reified/3, 3692
local_offset_reified/4, 3692
local_offset_with_resolution/3, 3686
local_offset_with_resolution/4, 3686
local_offset_with_resolution/5, 3686
local_optimization_problem_protocol, 2155
local_optimization_solver(Problem), 2158
local_tangent_plane/3, 511
local_tangent_plane_inverse/3, 512
local_time_type/2, 3683
local_time_type/3, 3683
local_time_type/4, 3682
local_time_type_reified/2, 3692
local_time_type_reified/3, 3691
local_time_type_reified/4, 3691
local_time_type_with_resolution/3, 3684
local_time_type_with_resolution/4, 3684
local_time_type_with_resolution/5, 3684
local_to_utc/3, 584
local_to_utc_tz/3, 594
local_to_utc_tz_with_resolution/4, 595
localhost_http_url/1, 2549
locate_directory/2, 675
locate_file/5, 675
locate_library/2, 674
location/2, 1112
lock_pack_facts/2, 2649
lock_registry_facts/3, 2648
locked_pairs/2, 2832
lof_anomaly_detector, 2166
log/3, 627
log_barrier(Problem,InnerSolver), 466
log_event/2, 2174
log_file/2, 2173
log_file_/2, 2170, 2172
log_point_/3, 615
logger, 2169
logging, 2171
logging/1, 2174
logging/3, 627
logging_to_file_/2, 2170, 2172
loggingp, 2172
logistic/3, 2819
logistic_normal/3, 2360
logistic_normal/4, 2361
logistic_normal_samples/4, 2362
logistic_normal_samples/5, 2362
logistic_regression_classifier, 2176
lognormal/3, 2815
logout_url/4, 2531, 2563
logseries/2, 2817
logtalk, 483
logtalk_packs/0, 2656
logtalk_packs/1, 2656
long_flags/1, 449
longest_common_increasing_subsequence/3, 3342
longest_common_subsequence/3, 3318, 3340
longest_common_subsequence_length/3, 3318
longest_common_substring/3, 3318
longest_decreasing_subsequence/2, 3341
longest_increasing_subsequence/2, 3341
longest_prefix/4, 3551
longest_repeating_subsequence/2, 3342
lookup/2, 747
lookup/3, 747, 2186, 3549
lookup/4, 747
lookup_in/3, 2441
lookup_prefix/4, 3550
loop, 2177
loopp, 2178
lovins_stemmer(Representation), 3230
low_surrogate/1, 316

lower_case/2, 3392
lower_triangular_part/2, 2105
lower_triangular_part/3, 2106
lower_upper/2, 3590
lowercase_ascii_codes/2, 1329
lpop/3, 2957
lpush/4, 2957
lrange/5, 2958
lrem/5, 2959
lru_cache, 2183
lru_cache_protocol, 2184
lt_language_profile, 1927
ltrim/5, 2959
lucas/2, 3626
lv_language_profile, 1928

M

magic, 3800
magic/2, 3801
magic_expansion(Mode), 3801
magicise/4, 3800
mahalanobis_distance/4, 2351
mahalanobis_distance/5, 2352
make/1, 2017
make_directory/1, 2599
make_directory_path/1, 2599
make_set/3, 3709
man/1, 1202
manager_seed_/1, 1484
manager_state_/2, 1484
manhattan_distance/3, 3640
manhattan_norm/2, 3639
map/2, 640, 751, 833, 2266, 2581, 2794, 3775
map/3, 640, 751, 2266, 2794, 3648, 3764
map/4, 2267, 3764
map/5, 2267
map/6, 2267
map/7, 2268
map/8, 2268
map_both/3, 837, 3776
map_catching/2, 836, 3776
map_element/2, 1754
map_id/2, 285
map_impl/2, 764
map_impl/3, 764
map_invalid/2, 3776
map_member/3, 3826
map_or_else/3, 837, 2584, 3775
map_reduce/5, 2268
map_store/4, 3826
map_unexpected/2, 836
mapped_single_byte_character_set, 330
mapping/2, 331
mask32/1, 1144
mask64/1, 1149
massey_ranker, 2190
master_channel_frame_count/2, 232
match/1, 2843
materialize/0, 575
matrix_column/3, 2101
matrix_column_means/2, 2097
matrix_diagonal/2, 2102
matrix_diagonal/3, 2102
matrix_entry/4, 2835
matrix_matrix_product/3, 2096
matrix_rank/2, 2109
matrix_rank/3, 2109
matrix_row/3, 2100
matrix_row_means/2, 2096
matrix_trace/2, 2104
matrix_value/4, 2101
matrix_vector_product/3, 2096
max/2, 3218, 3609, 3636
max/3, 749
max_clauses/1, 2276
max_flow/5, 942
max_flow_common(Dictionary), 939
max_flow_protocol, 941
max_inv_preds/1, 2277
max_path/5, 1076
max_probability/3, 366
max_size/1, 64
max_tree/3, 1106
maxheap, 1193
maximal_cliques/2, 1083
maximize/5, 940
maximum_cliques/2, 1084
maximum_linear_system_residual/4, 2990
maybe, 2572
maybe/0, 2812
maybe/1, 2812
maybe/2, 2812
maybe_call/1, 2813
maybe_call/2, 2813
mcp_cache_protocol, 2192
mcp_completion_protocol, 2193
mcp_multiround_protocol, 2195
mcp_prompt_protocol, 2197
mcp_resource_protocol, 2199
mcp_server, 2202
mcp_server_2025_06_18_spec, 2205
mcp_server_2025_11_25_spec, 2212
mcp_server_2026_07_28_spec, 2214
mcp_server_application, 2218
mcp_server_spec_protocol, 2225
mcp_server_stdio_transport, 2228
mcp_server_streamable_http_transport, 2229
mcp_server_transport_protocol, 2236
mcp_streamable_http_handler, 2238
mcp_tool_protocol, 2239
mcp_ui_protocol, 2243
md5, 1159

mean_absolute_error/3, 3469
mean_absolute_percentage_error/3, 3469
mean_center/2, 1015
mean_deviation/2, 3221
mean_squared_error/3, 3229
mean_values/2, 870, 2935
measurement/4, 2845
media_type/2, 1376
media_types/1, 1287
median/2, 3220, 3638
median_deviation/2, 3222
meets/2, 1669
member/2, 1676, 2807, 3106, 3610
memberchk/2, 3106, 3610, 3665
memcached, 2245
merge/3, 1189
merge_options/2, 2591, 3818
merged_options/2, 1577, 1598, 1599
mermaid_graph_language, 727
message/3, 1553
message_body/2, 20
message_cache_/1, 1999, 2000, 2056, 2058--2060
message_connection_tokens/2, 1316
message_delivery_tag/2, 21
message_diagram_description/1, 678
message_exchange/2, 21
message_has_connection_token/2, 1316
message_header_values/3, 1316
message_hook/4, 486
message_pack(StringRepresentation), 2257
message_prefix_file/6, 486
message_prefix_stream/4, 486
message_properties/2, 20
message_property/3, 20
message_routing_key/2, 21
message_tokens//2, 486
met_by/2, 1669
meta, 2259
meta/1, 450
meta_compiler, 2269
meta_type/3, 3659
metagol, 2271
metagol_example_protocol, 2279
metap, 2260
metaphone/2, 3321
metaphone_match/2, 3321
metarule/6, 2274
metarule_next_id/1, 2277
method/1, 1322
method/2, 1301, 1802
method_/8, 1994
method_not_found/2, 1798
method_priority/2, 1646
mget/3, 2252, 2949
mi_metric, 426
mi_metric(Stroud), 427
mibenum/1, 318
middleware/2, 1430
midpoint/3, 1013
milliseconds/2, 301
milp_branch_and_bound, 2134
mime_types, 2281
min/2, 3218, 3610, 3636
min/3, 749
min_clauses/1, 2276
min_cost_flow/4, 947
min_cost_flow/5, 947
min_cost_flow_protocol, 946
min_cost_max_flow/6, 948
min_cost_max_flow/7, 948
min_cut/5, 943
min_cut_value/4, 938
min_cuts/5, 945
min_distances/3, 1076
min_max/3, 3218, 3636
min_max_normalization/2, 3228
min_max_normalize/4, 3214
min_path/5, 1076
min_path_bellman_ford/5, 1102
min_paths/3, 1097
min_predecessors/3, 1077
min_tree/3, 1106
mine/2, 2733
mine/3, 106, 2733
mine/4, 106
minheap, 1194
minimal_output, 2045
minimum_enclosing_circle/3, 1015
minizinc, 2288
minizinc_expansion, 2289
minizinc_output, 2290
minizinc_problem_protocol, 2292
minizinc_process, 2293
minizinc_program, 2296
minizinc_protocol, 2298
minizinc_statement/2, 2293
minkowski_distance/4, 3640
missing_one_hot_encode/3, 2980
missing_predicate_directive_/3, 3819
mixed_feature_distance/5, 357
mlp_classifier, 2305
model/2, 91, 2303
model_binding/2, 2493, 3153
model_diagnostics/8, 1959
model_value/3, 2493, 3154
modes/2, 3221, 3638
modified_z_score_anomaly_detector, 2306
module_predicate_called_/3, 3819
module_property/2, 730
modules_diagram_support, 729
monitor, 816
monitor/1, 814

monitor/4, 814
monitor_activated/0, 818
monitored/1, 814
monitoring, 496
monitorp, 817
monitors/1, 813
month/2, 299
month_weekday_date/5, 585
mqtt, 2308
mqtt_property/3, 2326
mr_language_profile, 1929
mrmr_feature_selector, 2328
ms_language_profile, 1930
msd_similarity, 2906
msd_similarity/3, 2920
mset/3, 2950
msort/2, 3611
msort/3, 3611
mul32/3, 1146
mul64/3, 1149
multi_classification_ripple_down_rules, 3016
multi_source_max_flow/5, 942
multifile_directive_/3, 3820
multinomial/2, 3630
multinomial/3, 2366
multinomial_density/4, 2368
multinomial_log_density/4, 2369
multinomial_quantile/4, 2370
multinomial_samples/4, 2367
multiplayer_dataset_connected_components/2, 2873
multiplayer_dataset_declared_items/2, 2871
multiplayer_dataset_declared_matches/2, 2871
multiplayer_dataset_events/2, 2872
multiplayer_dataset_items/2, 2871
multiplayer_dataset_match_teams/3, 2872
multiplayer_dataset_matches/2, 2872
multiplayer_dataset_summary/2, 2873
multiplayer_ranking_dataset_protocol, 2842
multiset/3, 2332
multiset/4, 2332
multiset_index/4, 2335
multiset_index/5, 2335
multisets, 2329
multisets/3, 2332
multisets/4, 2332
multisets_protocol, 2330
multivariate_distributions(Random), 2339
multivariate_distributions_protocol, 2340
multivariate_normal/3, 2342
multivariate_normal/4, 2342
multivariate_normal_density/4, 2345
multivariate_normal_density/5, 2346
multivariate_normal_log_density/4, 2348
multivariate_normal_log_density/5, 2348
multivariate_normal_samples/4, 2343
multivariate_normal_samples/5, 2344
multivariate_t/4, 2353
multivariate_t/5, 2354
multivariate_t_density/5, 2356
multivariate_t_density/6, 2357
multivariate_t_log_density/5, 2358
multivariate_t_log_density/6, 2359
multivariate_t_samples/5, 2355
multivariate_t_samples/6, 2355
murmurhash3_x64_128, 1160
murmurhash3_x86_128, 1162
murmurhash3_x86_32, 1163
must_precede/3, 3124
must_precede_or_equal/3, 3125
mutate/2, 975
mutation/2, 2400
mutation/3, 2418, 2419
mutation/4, 2420
mutation_rate/4, 977
mutation_testing, 2380
mutation_testing_messages, 2391
mutations, 2417
mutations_store, 2418
mutator_common, 2392
mutator_protocol, 2399
mutual_information_feature_selector, 2421
mutual_information_score, 896
mutual_information_score(Discretization), 897

N

n_grams(Representation), 2422
n_grams/3, 2424
n_grams/4, 2425
n_grams_protocol, 2423
nack/3, 3239
naive_bayes_classifier, 2430
naive_forecast/3, 3470
name/1, 49, 318, 448, 2617, 2674
name_of_day/3, 581
name_of_month/3, 581
named_entity/2, 3385
nan_bits/1, 1608
nan_kind/2, 1614
nan_payload/2, 1613
nan_representation/1, 1619
nanoid, 2431
nanoid(Representation,Size,Alphabet), 2432
nanoid_protocol, 2433
natural, 3625
natural//1, 1054
navltree, 2437
nbintree, 2438
nearest_centroid_classifier, 2434

nearest_coordinate/5, 1015
nearest_point_on_polyline/4, 1023
nearest_point_on_segment/4, 1023
neighbor_state/2, 3133, 3359
neighbor_state/3, 3134, 3359
neighbors/2, 989, 3360
neighbors/3, 1073
neighbors_bits/3, 995
neighbors_int/3, 993
nelder_mead(Problem), 2163
nested_dictionary_protocol, 2439
network/1, 1658
new/1, 117, 1069, 1745, 2065, 2440, 3018, 3546, 3652
new/2, 118, 925, 1070, 1658, 1744, 2066, 2185, 3018, 3115, 3709
new/3, 925, 1070, 1668
new_hash_state/1, 1158
new_line//0, 1047
new_lines//0, 1048
new_matrix/4, 2094
new_problem/1, 2125
new_vector/3, 2091
new_vector_like/2, 2091
newton_root_finder(Function), 2464
next/2, 3837
next/3, 3838
next/4, 749, 1525
next_arrangement/3, 73
next_cartesian_tuple/3, 208
next_combination/3, 446
next_derangement/3, 651
next_distinct_partition/3, 1636
next_distinct_partition/4, 1636
next_impl/5, 765
next_multiset/3, 2338
next_occurrence/1, 2396
next_partition/3, 1634, 2697
next_partition/4, 1635, 2698
next_permutation/2, 2748
nextto/3, 3611, 3665
ngram_language_detector, 1931
nl_language_profile, 1932
nmea, 2444
nmea_protocol, 2445
nmf_projection, 2449
no_content_response/2, 3008
no_language_profile, 1933
noc_metric, 428
node/2, 122
node/3, 1763
node/6, 672
node/7, 714
node2_from_node4/2, 765
node_/6, 678
node_capacity_max_flow/6, 943
node_counter_/1, 657
node_id_/2, 658
node_path_/2, 678
nodebug/0, 621
nodes/1, 45
nodes/2, 123, 1659
nodes/3, 1785
nogood/2, 125
nolog/3, 627
nologall/0, 628
non_blank//1, 1048
non_blanks//1, 1049
non_standard_predicate_call_/2, 3820
non_terminal_abbreviation/1, 3519
nonempty_subsequence/2, 3336
nonempty_subsequences/2, 3335
nor_metric, 429
normal/3, 2815, 3715
normal_density/4, 3718
normal_distribution/4, 3720
normal_element/2, 1246
normal_log_density/4, 3719
normal_quantile/4, 3720
normal_samples/4, 3716
normalize/2, 1676, 3754
normalize_coefficient_grid/2, 1642
normalize_continuous/4, 378, 2979
normalize_cookie_attributes/2, 1282
normalize_coordinate/2, 1008
normalize_date_time/2, 586
normalize_json_pairs/2, 1313
normalize_json_value/2, 1313
normalize_missing_series/3, 3463
normalize_missing_values/4, 3476
normalize_options_term/2, 2547
normalize_polygon_orientation/3, 1018
normalize_range/2, 3641
normalize_range/4, 3641
normalize_scalar/2, 3642
normalize_scores/2, 1908
normalize_strengths/6, 2854
normalize_unicode/3, 3389
normalize_unicode_codes/3, 3401
normalize_unit/2, 3642
normalize_values/2, 2933
normalize_vector/2, 2114, 2933
normalize_vector/3, 2115
normalize_whitespace/2, 3396
normalize_whitespace/3, 3397
normalize_whitespace_codes/6, 3402
normalized_adjacency/5, 3707
nospy/1, 624
nospy/3, 625
nospy/4, 626
nospyall/0, 626
not64/2, 1151
not_excluded_file/3, 413

not_excluded_file/4, 674
note/1, 2017
note/2, 2675
note/3, 2618
notification/2, 1796
notification/3, 1796
notify/1, 2204, 2227, 2237
notrace/0, 622
now/3, 591
nrbtree, 2443
nth0/3, 3612, 3666
nth0/4, 3612, 3666
nth1/3, 3612, 3666
nth1/4, 3613, 3667
nth_arrangement/4, 69
nth_arrangement/5, 70
nth_cartesian_tuple/3, 205
nth_cartesian_tuple/4, 205
nth_combination/4, 442
nth_combination/5, 442
nth_derangement/3, 646
nth_derangement/4, 647
nth_distinct_arrangement/4, 71
nth_distinct_cartesian_tuple/3, 206
nth_distinct_combination/4, 444
nth_distinct_derangement/3, 649
nth_distinct_multiset/4, 2336
nth_distinct_partition/3, 1630, 2692
nth_distinct_partition/4, 1630, 2693
nth_distinct_partition/5, 1630, 2693
nth_distinct_permutation/3, 2750
nth_multiset/4, 2334
nth_multiset/5, 2334
nth_partition/3, 1628, 2690
nth_partition/4, 1628, 2690
nth_partition/5, 1628, 2690
nth_permutation/3, 2748
nth_permutation/4, 2749
null/1, 1749
null_device_path/1, 2602
null_redirect/2, 1727
null_space/2, 2113
null_space/3, 2114
number, 3630
number//1, 1055
number_grammars(Format), 1051
number_of_edges/2, 1075
number_of_tests/1, 2007
number_of_vertices/2, 1074
number_string/2, 3324
numberlist, 3633
numberlistp, 3635
numbervars/1, 3655
numbervars/3, 3654
nysiis/2, 3320
nysiis_match/2, 3320

O

object_file_/2, 1997
object_predicate_called_/3, 3818
object_wrapper_hook, 1233
object_wrapper_hook(Name,Relations), 1235
object_wrapper_hook(Protocol), 1234
objective/2, 2156
objective/3, 2495
objective/4, 2127
objective_direction/2, 463
objective_value/2, 2130
observation/2, 3494
observed_series/2, 844
observer, 631
occurrences/2, 3614
occurrences/3, 3614
occurs/2, 3652
ocf/2, 220, 224
ocf_flag/2, 232
ode_solver(System), 2465
ode_solver_protocol, 2467
ode_system_protocol, 2470
of/2, 2575
of_expected/2, 827
of_invalid/2, 3767
of_invalids/2, 3767
of_unexpected/2, 827
of_valid/2, 3767
offset/2, 3680
offset/3, 3455, 3680
offset/4, 3679
omit_path_prefix/3, 677
omt, 2486
omt_expansion, 2488
omt_problem_protocol, 2489
omt_protocol, 2490
one_class_svm_anomaly_detector, 2496
one_hot_encode/4, 2980
one_or_more//0, 1059
one_or_more//1, 1058
one_or_more//2, 1058
opcode/2, 1547
open/1, 1459, 1467, 1477, 1579, 3048
open/2, 1459, 1467, 1477, 1508, 1530, 3048
open/3, 1337, 1438, 1509, 1530
open/4, 1337, 1438, 1559
open/5, 1560
open/8, 3159
open_ai, 2497
open_ai_api, 2499
open_ai_backend_protocol, 2500
open_ai_catalog, 2501
open_ai_client, 2503
open_ai_event_stream, 2505
open_ai_server(Backend), 2507
open_api, 2511

open_api_bad_request_response/1, 1415
open_api_info/1, 1432, 2508
open_api_parameters/2, 1414
open_api_provider_protocol, 2522
open_api_request_body/3, 1415
open_api_request_validation_error_response/4, 3009
open_api_response_validation_error_response/5, 3010
open_api_schema/2, 1433, 2509
open_api_security/1, 1433, 2508
open_api_security_scheme/2, 1433, 2509
open_api_servers/1, 1432, 2508
open_api_validate_request/1, 2510, 3009
open_api_validate_response/1, 2510, 3009
open_connection/4, 1494
open_connection_pool/4, 1496
open_id, 2525
open_id_client, 2535
open_id_der, 2540
open_id_discovery, 2542
open_id_helpers, 2545
open_id_jwks, 2553
open_id_jwks_cache, 2555
open_id_jwt, 2559
open_id_logout, 2562
open_id_openssl, 2564
open_id_pkce, 2565
open_id_response, 2568
open_listener/4, 1492
open_session/4, 1519, 1539
open_session/5, 1520, 1539
open_sse/4, 1273
open_websocket/4, 1272
operating_system_machine/1, 2609
operating_system_name/1, 2609
operating_system_release/1, 2610
operating_system_type/1, 2609
operation/3, 2514
operation/8, 2502
operation_count/1, 2498, 2502
operation_properties/2, 2503
operation_url/4, 2505
operations/1, 1429, 2523
optics_clusterer, 2570
optimization_bounds/6, 847
optimization_components/8, 847
optimization_initial_point/6, 847
optimize_rule_body/2, 571
optimize_strengths/8, 2840, 2851
option/2, 2590
option/3, 2590
option_arguments/3, 1728
optional, 2574
optional(Optional), 2578
options, 2586
options_protocol, 2587
or/2, 838, 2581, 3777
or64/3, 1150
or_else/2, 834, 2582, 3777
or_else_call/2, 835, 2583, 3778
or_else_fail/1, 835, 2583, 3778
or_else_get/2, 834, 2582, 3778
or_else_throw/1, 835, 3779
or_else_throw/2, 838, 2583, 3779
order/1, 1614, 1618
ordered_scores/3, 2887
origin_endpoint/2, 1319
originator/1, 52
orphaned/0, 2627
orphaned/2, 2626
os, 2592
os_types, 2593
osp, 2595
otp, 2611
otp_protocol, 2612
out/1, 2072
out/2, 2072
out_degree/3, 1063
outdated/0, 2626
outdated/1, 2626
outdated/2, 2625
outdated/4, 2624
outdated/5, 2624
outer_product/3, 2098
output_edges/1, 673
output_externals/1, 670
output_file/4, 671
output_file_name/2, 712
output_file_path/4, 674
output_files/2, 671
output_library/3, 670
output_missing_externals/1, 673
output_node/6, 672
output_rdirectory/3, 670
output_rlibrary/3, 669
output_schema/2, 2242
output_stream_/1, 2216
output_sub_diagrams/1, 671
overlapped_by/2, 1670
overlaps/2, 1670
overlay_http_headers/3, 1396
overlay_http_properties/3, 1396
ows_code/1, 1329

P

pack_dependency/6, 2632
pack_metadata/4, 2628
pack_object/3, 2630
pack_property/4, 2629
pack_protocol, 2616
package/1, 51

packet_order_flag/2, 234
packet_type/2, 2311
packetize_aos_frames/7, 272
packetize_aos_frames/8, 273
packetize_aos_packets/6, 270
packetize_aos_packets/7, 271
packetize_tm_frames/7, 271
packetize_tm_frames/8, 272
packetize_tm_packets/6, 269
packetize_tm_packets/7, 269
packets/2, 251
packs, 2619
packs_common, 2649
packs_messages, 2660
packs_specs_hook, 2661
pad_md/4, 1147
pad_md_tail/3, 1152
pad_md_tail/5, 1147
pae/2, 2714
page_fragment_reply/4, 1369
page_fragment_reply/5, 1369
pair_key_value/3, 1312, 1833, 2552
paired_predictions/4, 1645
pairing_heap(Order), 1195
pairing_heap_max, 1196
pairing_heap_min, 1197
pairs, 3645
pairs_to_curly/2, 2224
pairs_to_edges/2, 1067
pairs_to_object/2, 1311
pairwise_dataset_connected_components/2, 2866
pairwise_dataset_declared_items/2, 2865
pairwise_dataset_items/2, 2865
pairwise_dataset_matchups/2, 2866
pairwise_dataset_preferences/2, 2865
pairwise_dataset_summary/2, 2866
pairwise_dataset_win_totals/2, 2866
pairwise_measurement_dataset_connected_components/2, 2868
pairwise_measurement_dataset_declared_items/2, 2867
pairwise_measurement_dataset_items/2, 2867
pairwise_measurement_dataset_measurements/2, 2867
pairwise_measurement_dataset_protocol, 2844
pairwise_measurement_dataset_summary/2, 2868
pairwise_ranking_dataset_protocol, 2846
pairwise_strength_ranker_common, 2847
parameter/3, 1414
parameter_bounds/2, 846
parameters/3, 1413
parameters_from_point/3, 846
params/2, 1802
parent/1, 1220
parent/2, 991
parent_stack_/1, 657
parenthesis/2, 3590
parents/1, 1220
parse/2, 94, 128, 131, 133, 135, 137, 139, 141, 192, 211, 217, 277, 305, 504, 1001, 1039, 1544, 1616, 1760, 1765, 1770, 1777, 1784, 1790, 1810, 2258, 2446, 2513, 2785, 3502, 3525, 3537, 3751, 3809, 3824, 3831
parse/3, 128, 312, 504, 2446, 2785, 3752, 3824
parse/4, 452, 1374
parse/5, 453
parse_all/2, 3831
parse_authentication_info/2, 1348
parse_authentication_parameters/3, 1286
parse_authorization/2, 1252, 1347, 1385
parse_body/4, 1298
parse_challenge/2, 1250, 1346, 1384
parse_cookie/2, 1280
parse_domain/2, 2739
parse_domain/3, 2739
parse_error/1, 1798
parse_fragment/2, 1791
parse_frame/3, 237
parse_frames/3, 238
parse_headers/2, 1297
parse_lines/4, 3503
parse_problem/2, 2739
parse_problem/3, 2740
parse_relative/2, 1791
parse_request/2, 1292
parse_request_line/2, 1295
parse_response/2, 1293
parse_scope_value/2, 1395
parse_set_cookie/4, 1281
parse_status_line/2, 1296
parse_token/4, 2715
part/4, 1377
part_body/2, 1378
part_headers/2, 1378
part_properties/2, 1379
partial_/1, 2053, 2055
partial_map/4, 754
particle_swarm_optimization(Problem), 2676
particle_swarm_optimization(Problem,Random), 2677
particle_swarm_optimization_protocol, 2680
partition/2, 1623, 2686
partition/3, 824, 1623, 2686, 3764
partition/4, 1624, 2263, 2687, 3100
partition/5, 3614
partition/6, 2263
partition_body_literals/5, 571
partition_index/3, 1629, 2691
partition_index/4, 1629, 2691
partition_index/5, 1629, 2691

partition_number/2, 3629
partitions, 2683
partitions/2, 1623, 2686
partitions/3, 1623, 2686
partitions/4, 1624, 2687
partitions_protocol, 2684
partner/2, 933
parts/2, 1376
paseto, 2699
paseto_claims, 2708
paseto_claims_helpers, 2711
paseto_helpers, 2713
paseto_keys, 2716
paseto_protocol, 2718
paseto_v4, 2727
pass_info/1, 803
pass_info/2, 803
passed_/3, 2042
password_hash/4, 528
password_hash_atom/2, 533
password_hash_needs_rehash/3, 530
patch/4, 1270
patch/5, 1271, 1342, 1464
path/3, 1075
path_concat/3, 2598
path_consistency/2, 1665
path_ids/4, 934
path_parameter/3, 3002
path_template_parameters/2, 1434
path_within_root/2, 1306
paths/3, 1785
pattern_length/2, 3095
pattern_miner_common, 2729
pattern_miner_diagnostics/6, 2730
pattern_miner_diagnostics_data/2, 2730
pattern_miner_export_template/4, 2731
pattern_miner_options/2, 2734
pattern_miner_protocol, 2732
payload/2, 1548, 1556
payload_hash/2, 3059
pbkdf2/6, 526
pca_projection, 2737
pcdata_7bit//1, 3828
pddl, 2738
pearson_similarity, 2907
pearson_similarity/3, 2919
peek_algorithm/2, 1815
peek_back/2, 639
peek_front/2, 639
peek_key_id/2, 1815, 2707
penalty_sub_problem(Problem,OuterIteration,Start/5, 2761, Rho,Sign), 467
pending_data/2, 249
pending_fragments/2, 240, 250, 288
pending_packets/2, 268
percentile/3, 3225
period/1, 2889
permutation/2, 2744, 2810, 3615, 3667
permutation/3, 2745
permutation_index/3, 2749
permutation_index/4, 2749
permutations, 2742
permutations/2, 2744
permutations/3, 2745
permutations_protocol, 2743
persist/3, 2953
phase1_sub_problem(Problem,StartPoint), 468
pid/1, 2596
piecewise_costs, 949
piecewise_costs(Dictionary), 950
piecewise_linear_interpolator, 2471
pin/0, 2652
pin/1, 2652
ping/2, 2314
pinned/1, 2653
pkcs7_pad/3, 153
pkcs7_unpad/4, 154
pl_language_profile, 1934
plackett_luce_common, 2852
plackett_luce_last_ranker, 2753
plackett_luce_ranker, 2754
plantuml_graph_language, 731
pls_projection, 2756
plus/3, 3600
point_in_polygon/2, 1016
point_to_polyline_distance/3, 1023
poisson/2, 2820
polygon_area/2, 1016
polygon_bounding_box/2, 1017
polygon_centroid/2, 1017
polygon_covering/4, 996
polygon_orientation/2, 1018
polygon_perimeter/2, 1025
polygon_perimeter/3, 1026
polygons_intersect/2, 1026
polyline_covering/4, 997
polyline_length/2, 1024
polyline_length/3, 1024
polyline_resample/3, 1025
polyline_simplify/3, 1024
polyline_split_at_distance/4, 1025
pool_config/5, 27
pop_back/3, 638
pop_front/3, 638
population, 3208
population/4, 901
port_/5, 2762
porter_stemmer(Representation), 3231
portray_clause/1, 2146
ports_profiler, 2757
position_bounds/1, 774, 2157, 2681

positive_max/2, 463
positive_series/1, 3430
positive_smoothing/4, 1649
possible/4, 1660
post/4, 1267
post/5, 1268, 1341, 1463
postorder/2, 741
power/2, 2820
power_sequence/4, 3601
power_set/2, 3336
powerset/2, 3106
powershell_command/2, 1725
pp/1, 3825
pp_string/1, 3827
pprint/1, 2274
pprint_clause/1, 2278
pq_insert/3, 1100
precision/1, 1614, 1618
precision/3, 988
precision_spec/5, 1609
pred_info/3, 802
predicate/1, 654
predicate/2, 604, 654, 2383
predicate/3, 605, 2384
predicate_breakpoint_/3, 616
predicate_called_but_not_defined_/2, 3818
predicate_coverage_/4, 2047
predicate_coverage_/8, 2001
predicate_directive_suppression(Entity,Predicate,Directive,Context), 2401
predicate_entity_/4, 1980
predicate_info_pair_score_hook/4, 420
predicate_info_score_hook/3, 420
predicate_mode_score_hook/3, 419
predicate_mode_score_hook/5, 420
predicate_mutants/3, 2387
predicate_mutants/4, 2387
predicate_stratum/3, 577
predicate_stratum_/3, 566
predicates/2, 604
predicates/3, 604
predicates_in_stratum/2, 569
predict/3, 37, 359, 2992
predict/4, 38, 1860, 2436
predict_distribution/3, 144, 968
predict_from_probabilities/3, 366
predict_probabilities/3, 367
predict_probabilities/4, 1861, 2436
predict_scores/3, 1967, 2788
preference/3, 2847
preferred_mime_name/1, 318
prefix/0, 2657
prefix/1, 2656
prefix/2, 3575, 3615, 3667
prefix/3, 3615
prefix_data/2, 251
prefix_span_pattern_miner, 2763
prefixes/3, 3551
preflight_response/3, 1333
preorder/2, 741
prepare/2, 2226, 2231
prepare_cipher/4, 151
prepare_feature_columns/7, 864
prepare_key/2, 155
prepare_learning/4, 1642, 3428
prepare_parameters/4, 1648
prepare_points/3, 2460
prepare_seasonality/8, 3429
prepare_training_rows/6, 377
prepare_training_rows/7, 377
prepend/3, 2249
preprocess_groups/10, 2853
preprocess_matchups/6, 2849
preprocessing_diagnostics/3, 782
presigned_get_object/4, 3041
presigned_get_object/5, 3052
presigned_post_object/4, 3044
presigned_post_object/5, 3054
presigned_put_object/4, 3042
presigned_put_object/5, 3053
presigned_query_pairs/7, 3061
previous/2, 3838
previous/3, 3838
previous/4, 748
previous_cartesian_tuple/3, 209
previous_combination/3, 446
previous_derangement/3, 651
previous_distinct_partition/3, 1636
previous_distinct_partition/4, 1637
previous_impl/5, 765
previous_multiset/3, 2338
previous_partition/3, 1635, 2698
previous_partition/4, 1635, 2698
previous_permutation/2, 2748
primal_dual_interior_point(Problem), 469
primary_office_bic/1, 1717
print_anomaly_detector/1, 39
print_anomaly_detector_template/1, 34
print_association_rule_miner/1, 109
print_classifier/1, 362
print_classifier_template/1, 351
print_clusterer/1, 384
print_dimension_reducer/1, 792
print_dimension_reducer_details/3, 787
print_dimension_reducer_properties/1, 779
print_flags/0, 910, 914
print_flags/1, 910
print_forecaster/1, 3493
print_forecaster_template/1, 3460
print_goal_hook, 1236
print_message/3, 485

print_message_token/4, 485
print_message_tokens/3, 485
print_mutation/3, 2393
print_pattern_miner/1, 2736
print_problem/1, 2131
print_program/1, 91, 2303, 2492, 2779, 3085, 3153, 3198
print_ranker/1, 2862
print_ranker_template/1, 2857
print_readme_file_path/1, 2657
print_recommender/1, 2930
print_recommender_template/1, 2913
print_regressor/1, 2995
print_regressor_template/1, 2970
print_result/1, 92, 2304, 2494, 2780, 3087, 3155, 3199
print_results/1, 2494, 3156
print_selector/1, 890
print_selector_template/1, 878
print_solution/1, 2132
probabilistic_classifier_common, 365
probabilistic_classifier_protocol, 367
probabilistic_pca_projection, 2764
probability/3, 2779
probe_mutation_happened_/0, 2389
probing_/0, 2389
problem_from_matrices/8, 2128
problem_json_schema/1, 2520
problem_response/6, 3008
problem_response_descriptor/3, 2521
problog, 2765
problog_expansion, 2767
problog_output, 2768
problog_problem_protocol, 2769
problog_process, 2771
problog_program, 2773
problog_protocol, 2776
problog_statement/2, 2770
process, 2781
process_all/1, 406
process_connection_state_/3, 1421
process_directory/2, 405
process_entity/2, 404
process_file/2, 404
process_library/2, 405
process_listener_shutdown_requested_/1, 1424
process_listener_state_/5, 1422
process_rdirectory/2, 405
process_rlibrary/2, 406
product/2, 3217, 3637
product/3, 3107
profile/2, 1108, 1116, 1926
profile/3, 1109
profile_path/2, 1116
profile_response/3, 1109
profile_url/2, 1111
profile_weights/3, 185, 3424
program/1, 89, 2301, 2491, 2778, 3082, 3151, 3196
program/2, 90, 2302, 2492, 2778, 3083, 3152, 3197
program/3, 3084
program_to_clauses/2, 2273
progress/5, 46, 775, 976, 2157, 2682, 3135, 3362
progress_buffer_/1, 2217
progress_events_/1, 2234
progress_hook/5, 2163
progress_token_/1, 2234
project_components/4, 786
project_to_bounds/3, 2161
prolog_module_hook(Module), 1238
prompt/2, 1365
prompt_descriptors_to_json/2, 2219
prompt_get/3, 2198
prompt_get_round/4, 2196
prompts/1, 2198
pronouns/2, 1114
propagate/2, 1662
propagate/3, 1662, 3503
propagate/4, 3504
propagate/5, 3504
propagate_state/4, 3504
propagate_state/5, 3505
propagation_triple/2, 1664
propagation_triples/2, 1664
proper_prefix/2, 3616
proper_prefix/3, 3616
proper_subsequence/2, 3343
proper_suffix/2, 3623
proper_suffix/3, 3623
properties/2, 1548
property/1, 57, 1324, 1327
property/2, 1303, 1514, 1535, 1549
property/3, 2544
protect_request/4, 1252, 1350, 1387
proto_hierarchy, 1218
proto_hierarchyp, 1219
protobuf, 2784
prove/2, 3799
prove/3, 3799
provider/3, 2543
provider/4, 2543
provider_property/3, 2546
provider_property_default/4, 2547
provides/2, 2670
prune/3, 198
prune/5, 198
pseudo_inverse/2, 2112
pseudo_inverse/3, 2113
pseudo_random_protocol, 2800
pt_language_profile, 1935
public_key_pem/2, 1830, 1839, 2541
public_keypair/2, 2719
public_sign/3, 2722
public_sign/5, 2723

public_verify/3, 2724
public_verify/5, 2725
publish/4, 2312
push_back/3, 638
push_front/3, 637
put/4, 1269, 2187, 3014
put/5, 1269, 1342, 1464
put_object/6, 3038
put_object/7, 3055

Q

qda_classifier, 2787
qp_active_set, 470
qp_solver_protocol, 472
qr_decomposition/3, 2108
quadratic_penalty(Problem,InnerSolver), 473
quadrature(Function), 2472
quadrature_protocol, 2474
quartiles/4, 3225
quasi_skipping_/0, 613
query/1, 576
query/2, 576
query/3, 1786
query/4, 1271
query/5, 1272
query_atom/2, 2547
query_parameter/3, 3002
question_hook/6, 487
question_prompt_stream/4, 487
queue, 2788
queue_bind/4, 12
queue_declare/3, 11
queue_delete/3, 11
queue_purge/2, 13
queue_unbind/4, 12
queuep, 2789
quick_check/1, 2010
quick_check/2, 2010
quick_check/3, 2009
quote/3, 81, 2294, 2772, 3075, 3143, 3188
quoted_authentication_parameter/3, 1396

R

random, 2801
random(Algorithm), 2803
random/1, 2807
random/3, 2811
random_arrangement/3, 72
random_below/2, 523
random_bytes/2, 522
random_cartesian_tuple/2, 207
random_combination/3, 444
random_derangement/2, 649
random_distinct_arrangement/3, 73
random_distinct_cartesian_tuple/2, 208
random_distinct_combination/3, 445
random_distinct_derangement/2, 650
random_distinct_multiset/3, 2337
random_distinct_partition/2, 1633, 2696
random_distinct_partition/3, 1633, 2696
random_distinct_permutation/2, 2752
random_forest_classifier, 2826
random_forest_regression, 2827
random_individual/1, 974
random_multiset/3, 2337
random_node/1, 3761
random_partition/2, 1632, 2695
random_partition/3, 1632, 2695
random_permutation/2, 2751
random_projection, 2828
random_protocol, 2805
random_subsequence/2, 3347
random_tree/1, 3783
randomize/1, 2799, 2804
randseq/4, 2811
randset/4, 2811
range/2, 3219
range_query/5, 388
rank/3, 2859
rank_by_scores/3, 2883
rank_centrality, 2830
rank_correlation/3, 3226
ranked_pairs, 2831
ranker_common, 2855
ranker_diagnostics_data/2, 2856
ranker_export_template/4, 2856
ranker_options/2, 2861
ranker_protocol, 2858
ranker_scores_data/2, 2856
ranker_term_template/2, 2857
ranking_dataset_common, 2863
ranking_dataset_protocol, 2880
rating/3, 2909
rating_count/1, 2909
rating_dataset_protocol, 2908
rating_removal_records/3, 180, 1735, 3421
rating_scale/2, 2909
rbtree, 753
rd/1, 2075
rd/2, 2074
rd_list/2, 2076
rd_list/3, 2076
rd_noblock/1, 2075
rd_noblock/2, 2075
rdirectories/1, 1987
rdirectories/2, 1987
rdirectory/1, 399, 601, 665, 686, 1988, 3814
rdirectory/2, 399, 601, 665, 685, 1987, 3813
rdirectory/3, 665, 685
rdirectory_score/2, 403
reachable/3, 1073
read_file, 2740

read_file/2, 552, 2741, 3563
read_file/3, 550, 3561
read_file_by_line/2, 554, 3565
read_file_by_line/3, 553, 3564
read_frame/2, 1545
read_frame/3, 1546
read_framed_message/2, 1805
read_from_atom/2, 3366
read_from_chars/2, 3368
read_from_codes/2, 3370
read_message/2, 1554, 1805
read_message/4, 1589
read_message/5, 1590
read_message/6, 1590
read_mime_types/2, 2286
read_only_device_path/1, 2602
read_packet/2, 2317
read_request/2, 1452
read_response/2, 1276
read_response_head/2, 1277
read_session_message/5, 1595
read_stream/2, 552, 3563
read_stream/3, 551, 3562
read_stream_by_line/2, 555, 3566
read_stream_by_line/3, 553, 3564
read_term_from_atom/3, 3366
read_term_from_chars/3, 3367
read_term_from_chars/4, 3368
read_term_from_codes/3, 3369
read_term_from_codes/4, 3369
reader, 2890
readme/1, 2655
readme/2, 2655
readme_file_path/2, 2657
reason_code/3, 2312
reassemble_aos_frames/5, 264
reassemble_aos_frames/6, 265
reassemble_aos_frames/7, 266
reassemble_aos_packets/5, 262
reassemble_aos_packets/6, 263
reassemble_aos_packets/7, 264
reassemble_frames/6, 244
reassemble_frames/7, 245
reassemble_frames/8, 246
reassemble_packet_zone/5, 255
reassemble_packets/6, 242
reassemble_packets/7, 243
reassemble_packets/8, 244
reassemble_tc_frame/4, 290
reassemble_tc_frame/5, 290
reassemble_tc_frame/6, 291
reassemble_tc_frame_with_provenance/5, 294
reassemble_tc_frame_with_provenance/6, 294
reassemble_tc_frames/4, 292
reassemble_tc_frames/5, 292
reassemble_tc_frames/6, 293
reassemble_tc_frames_with_provenance/5, 295
reassemble_tc_frames_with_provenance/6, 296
reassemble_tm_frames/5, 259
reassemble_tm_frames/6, 260
reassemble_tm_frames/7, 261
reassemble_tm_packets/5, 257
reassemble_tm_packets/6, 257
reassemble_tm_packets/7, 258
rebuild_catalog_model/5, 3420
rebuild_catalog_model/6, 180
rebuild_feedback_model/7, 3422
rebuild_feedback_model/9, 182
receive/2, 1512, 1533
receive/3, 17, 1512, 1533, 2313, 3238
receive_binary/2, 1536
receive_data/2, 1519
receive_json/2, 1517, 1537
receive_term/2, 1518, 1538
receive_text/2, 1536
recommend/4, 2927
recommend_from_ratings/5, 2916
recommender_common, 2910
recommender_diagnostics_data/2, 2912
recommender_export_template/4, 2912
recommender_options/2, 2929
recommender_protocol, 2922
recommender_term_template/2, 2912
recommender_valid_data/1, 2912
record_/3, 2943
recorda/2, 2940
recorda/3, 2940
recorded/2, 2942
recorded/3, 2942
recorded_database, 2938
recorded_database_core, 2939
recorded_linter_warning_/7, 2041
recorded_warning_/4, 2143
recordz/2, 2941
recordz/3, 2941
redis, 2944
reference_/1, 2944
reference_kind/2, 3752
referenced_entity_/2, 702
referenced_logtalk_directory_/1, 694
referenced_logtalk_file_/1, 708
referenced_logtalk_library_/2, 724
referenced_module_/2, 702
referenced_predicate_/1, 738
referenced_prolog_directory_/1, 694
referenced_prolog_file_/1, 708
referenced_prolog_library_/2, 724
refine/5, 1661
refine_propagate/3, 1663
refine_propagate/4, 1664
refine_propagate/5, 1663
refine_propagate/6, 1663

refresh_token/4, 2529, 2538
register/2, 1580
registries, 2662
registry_loader_hook, 2672
registry_protocol, 2673
registry_seed_/1, 1584
registry_session_/3, 1585
registry_state_/2, 1584
regression_dataset_adapter(Dataset), 1962
regression_dataset_protocol, 2963
regression_examples_adapter(Declarations,Examples), 1964
regression_tree, 2996
regressor_common, 2965
regressor_diagnostics_data/2, 2968
regressor_export_template/4, 2968
regressor_options/2, 2994
regressor_protocol, 2991
regressor_term_template/2, 2968
regularization_grid_values/2, 1961
regularize_ridge_matrix/4, 2986
regularized_bradley_terry_ranker, 2997
relation/1, 1655
relation/3, 1668
relation/4, 1659, 3198
relation_set/1, 1675
relational_operator_inversion(Entity,Predicate,Clause,Index), 2402
relational_operator_replacement(Entity,Predicate,Clause,Index,Occurrence,PrintMutation), 2403
relative_standard_deviation/2, 3223
relativize/3, 3753
relax_neighbors/7, 1099
relax_problem/2, 2123
release/1, 25
release_date/1, 53
relevance/3, 2881
relief_feature_selector, 2999
relief_feature_selector_common, 898
relief_model/1, 899
relief_variant/1, 899
relieff_feature_selector, 3000
remember_included_directory/1, 693
remember_included_file/1, 707
remember_included_library/2, 722
remember_referenced_logtalk_directory/1, 693
remember_referenced_logtalk_file/1, 707
remember_referenced_logtalk_library/2, 723
remember_referenced_prolog_directory/1, 693
remember_referenced_prolog_file/1, 707
remember_referenced_prolog_library/2, 723
remote/3, 1031
remove/3, 1481, 3015
remove_candidate/3, 379
remove_catalog/3, 175
remove_constraints/3, 3121
remove_diacritics/2, 3389
remove_diacritics_codes/2, 3382
remove_directive_/2, 3822
remove_duplicates/2, 3616, 3668
remove_ratings/3, 173, 1734, 3416
remove_rule/1, 573
remove_time_points/3, 3116
removeDependent/1, 634
rename/4, 2953
replace_file/2, 2606
renew/2, 1482
repeated_labels/2, 1728
replace/3, 3841
replace/5, 2248
replace_content/3, 174, 1732, 3418
replace_diagnostic/4, 3466
replace_sub_atom/4, 3579
replay_series/5, 1641
reply/3, 1367
reply/4, 1368
reply_outcome_/1, 2216
report_directory/3, 2385
report_entity/3, 2384
report_final/5, 2163
report_library/3, 2385
report_mutation(Entity,Predicate,Clause,Index,Occurrence,PrintMutation), 2404
report_progress/6, 2162
repository_branch/1, 54
repository_commit/1, 55
repository_commit_abbreviated/1, 55
repository_commit_author/1, 56
repository_commit_date/1, 55
repository_commit_message/1, 56
representation/2, 118
request/3, 1796
request/4, 1262, 1796
request/5, 1263, 1339, 1461, 2504
request/6, 3028
request/7, 1290
request_body/2, 3003
request_body_bytes/4, 1299
request_cookies/3, 1469
request_cookies/4, 1470
request_endpoint/2, 1320
request_header/3, 3003
request_kind/2, 1364
request_listener_shutdown/1, 1440, 1494
request_origin/2, 1332
request_persistent/1, 1315
request_properties/2, 1367
request_property/2, 1367
request_shutdown/1, 1447, 1505, 1568
require_neighborhood/2, 3356
require_strong_connectivity/4, 2841, 2850

rescale/3, 3642
reset/0, 621, 672, 2141, 2287, 2400, 2651, 2760, 3789
reset/1, 2760
reset_counter/1, 2063
reset_counters/0, 2064
reset_flags/0, 908
reset_flags/1, 909
reset_genint/0, 980
reset_genint/1, 980
reset_gensym/0, 983
reset_gensym/1, 983
reset_monitor/0, 818
reset_seed/0, 2799, 2804
residual_fitted_values/3, 3464
residual_sum_of_squares/4, 2982
residual_totals/3, 3433
residuals/2, 1226
resize/2, 26
resolve/3, 3753
resolve_address/4, 2316
resolve_command_path/2, 2603
resolve_tenure/6, 3357
resolve_transport/3, 2316
resolved_target_path/3, 1306
resource_descriptors_to_json/3, 2219
resource_read/3, 2201
resource_read_round/4, 2196
resource_template_descriptors_to_json/2, 2220
resource_templates/1, 2201
resource_ui_meta/2, 2244
resources/1, 2200
response/3, 1403, 1797
response/4, 1404
response/6, 1291
response_body_bytes/4, 1299
response_middleware/2, 1430
response_persistent/1, 1315
response_producing/1, 3147
response_properties/2, 3031
rest, 3001
rest_action_error_response/4, 3010
rest_action_failure_response/3, 3010
restart_candidate/7, 3356
restart_state/2, 3361
restore/1, 2645
restore/2, 2644
restore_edb_facts/1, 567
restore_idb_facts/1, 567
restore_predicate_strata/1, 568
restore_seasonality/5, 3475
restore_snapshot/6, 567
restore_support_counts/1, 568
restore_support_edges/1, 568
result/2, 1803
result_property/2, 92, 2304, 2494, 2779, 3086, 3155, 3199
retract_batch/4, 122
retract_fact/1, 575
retract_justification/5, 120
retract_justifications/3, 120
retract_node/3, 121
retract_nodes/3, 121
reverse/2, 3617, 3668
revise/6, 3019
rewind/2, 3839
rewind/3, 3839
rewrite_diagnostics/4, 1650
rfc_metric, 431
rhumb_bearing/3, 1010
rhumb_destination_point/4, 1010
rhumb_distance/3, 1010
rhumb_midpoint/3, 1011
ridge_feature_statistics/3, 2983
ridge_regression, 3012
ripple_down_rules_common_protocol, 3017
ripple_down_rules_protocol, 3021
rising_factorial/3, 3627
rk45_ode_solver(System), 2477
rk4_ode_solver(System), 2478
rlibraries/1, 1985
rlibraries/2, 1984
rlibrary/1, 401, 603, 663, 684, 1985
rlibrary/2, 400, 602, 663, 683, 1985
rlibrary_score/2, 402
ro_language_profile, 1936
rol32/3, 1146
rol64/3, 1150
rollback/0, 574
root/2, 1030
root_finder(Function), 2479
root_finder_protocol, 2481
root_mean_squared_error/3, 3229, 3469
ror32/3, 1146
route/4, 1430
route_automatic_options_response/3, 1435
route_bad_request_response/3, 1435
route_distance/2, 1027
route_distance/3, 1027
route_distance/4, 1027
route_metadata/2, 1431
route_method_not_allowed_response/3, 1435
route_not_acceptable_response/3, 1436
route_not_found_response/2, 1434
route_open_api_request_body_example/2, 1431
route_open_api_response_example/2, 1432
route_parameter_declarations/2, 1417
route_parameter_extra_metadata/2, 1418
route_parameter_request_body_description/2, 1417
route_parameters/2, 1417

route_produces/2, 1434
row_values/4, 901
rows_to_design_matrix/3, 2982
rpop/3, 2958
rpush/4, 2957
rrelieff_feature_selector, 3023
ru_language_profile, 1937
rule/2, 3791
rule/3, 3790
rule/4, 3790
rule_/3, 565
rule_expansion(Mode), 3802
rule_kind/2, 104
rule_metric/3, 107
rules/1, 576
run/0, 2006
run/1, 2006
run/10, 3188
run/2, 43, 771, 971, 2006, 2159, 2678, 3131, 3354
run/3, 43, 771, 971, 2159, 2678, 3131, 3354
run/4, 43, 772, 972, 2160, 2679, 3131, 3355
run/7, 82, 2772, 3075, 3144
run/8, 2295
run_cli/3, 1727
run_quick_check_tests/5, 2015
run_session/3, 1571
run_session/4, 1572
run_session_connection/5, 1573
run_stdio_loop/3, 2215
run_test_set/0, 2015
run_test_sets/1, 2006
run_tests/0, 2015
run_tests/1, 2015
running_/0, 2216, 2233
running_test_sets_/0, 2039
runtime_type_/3, 2288

S

s3_client, 3024
s3_client(HTTPTransport), 3026
s3_endpoints, 3027
s3_headers, 3030
s3_protocol, 3032
s3_session(HTTPTransport), 3045
s3_signing, 3058
s3_xml, 3062
sadd/4, 2959
same_instant/2, 588
same_length/2, 3617, 3668
same_length/3, 3617
same_site/2, 1320
sample, 3209
sample/3, 2809
sample/4, 2809
sample_arrangements/4, 72
sample_cartesian_tuples/3, 207
sample_combinations/4, 445
sample_derangements/3, 650
sample_distinct_arrangements/4, 73
sample_distinct_cartesian_tuples/3, 208
sample_distinct_combinations/4, 445
sample_distinct_derangements/3, 650
sample_distinct_multisets/4, 2338
sample_distinct_partitions/3, 1634, 2697
sample_distinct_partitions/4, 1634, 2697
sample_distinct_permutations/3, 2752
sample_multisets/4, 2337
sample_partitions/3, 1632, 2695
sample_partitions/4, 1633, 2696
sample_permutations/3, 2751
sampling_protocol, 2814
sarif, 3066
sat, 3068
sat_expansion, 3069
sat_output, 3071
sat_problem_protocol, 3073
sat_process, 3074
sat_program, 3077
sat_protocol, 3079
save/0, 3817
save/1, 2643, 3676, 3816
save/2, 1472, 2642, 3676
save_dut1_entries/1, 3441, 3450
save_edge/5, 673
save_leap_seconds_entries/1, 3440, 3449
save_setup_terms/2, 2648
sbom, 3088
scalar_product/3, 3641
scale_matrix/3, 2099
scale_values/2, 871, 2932
scale_values/3, 871
scale_vector/3, 2092
scaled_row_difference/4, 2989
scan_left/4, 2264
scan_left_1/3, 2264
scan_right/4, 2265
scan_right_1/3, 2266
scard/3, 2961
schedule/2, 3117
schema/2, 1429, 2524
schulze_ranker, 3091
scope_directive_replacement(Entity,Predicate,Direct
2404
scopes/2, 1389
score/3, 38, 875, 1719
score/4, 2926
score_all/3, 38, 1719
score_all/4, 171, 1730, 3414
score_catalog_items/5, 1737, 3423
score_catalog_items/6, 183
score_columns/3, 2422
score_columns/4, 348

score_content/4, 171, 1731, 3415
score_features/4, 881
score_observation/7, 1648
score_query_items/4, 183
score_ranker_common, 2882
score_ranker_data/4, 2886
score_ranker_label/1, 2884
score_ranker_model/1, 2884
score_ranker_model_common, 2883
score_ranker_term/4, 2885
score_seasonal/9, 3431
scores/2, 2860
scores/3, 1912
script_language_detector, 1938
script_ratios/2, 1910
scripts_to_languages/2, 1911
sdbm_32, 1164
sdbm_64, 1165
search/1, 2632
search_arguments/5, 1722
search_command/4, 1722
search_index_cell_size/2, 389
search_indexing, 386
seasonal_accumulate_cycle/4, 3486
seasonal_autocorrelation_test/4, 3473
seasonal_bucket_means/2, 3481
seasonal_center_even/2, 3480
seasonal_centered_values/5, 3478
seasonal_check_buckets/2, 3487
seasonal_check_method/1, 3479
seasonal_complete_autocorrelation_test/4, 3483
seasonal_covariance/4, 3479
seasonal_deviations/4, 3481
seasonal_fold_cycles/3, 3481
seasonal_gap_centered/4, 3484
seasonal_gap_covariance/4, 3485
seasonal_gap_deviations/4, 3486
seasonal_gap_even/2, 3485
seasonal_gap_lags/7, 3484
seasonal_gap_windows/5, 3485
seasonal_inverse_factors/3, 3482
seasonal_lag_scores/7, 3478
seasonal_naive_forecast/4, 3472
seasonal_normalize/4, 3482
seasonal_positive_values/1, 3479
seasonal_restore_values/5, 3483
seasonal_window_means/5, 3480
secant_root_finder(Function), 2484
secondary_header/2, 235, 281
secondary_header_flag/2, 233, 279
secondary_header_time/3, 281
section_started_/1, 656
section_visibility/2, 1114
secure_compare/2, 524
security/1, 1428, 2523
security_scheme/2, 1429, 2524
seed_/2, 2799, 2805
seen_/1, 2399
segment_data/2, 287
segment_header/2, 230
segment_header_suffix/2, 286
segment_header_suffixes/2, 287
segment_length_identifier/2, 234
segments/5, 1827
select/3, 2807, 3107, 3618, 3669
select/4, 766, 2808, 3618
select_above_threshold/3, 882
select_active/3, 1959
select_alphas/7, 1643
select_betas/7, 1644
select_candidate/10, 3356
select_frequent_item_supports/3, 963, 3096
select_interval/3, 2461
select_key/3, 1841, 2554
select_key/4, 1842
select_keys/4, 1842, 2717
select_methods/7, 1643
select_metric_pivot/4, 389
select_pivot_row/3, 2988
select_top_k/3, 882
selectchk/3, 3107, 3618
selectchk/4, 3619
selected_features/2, 887
selected_test_/1, 2040
selection_exposures/4, 2854
selector_diagnostics_data/2, 878
selector_export_template/4, 878
selector_options/2, 889
selector_term_template/2, 878
send/2, 1511, 1532
send/3, 1511, 1532, 1582, 2947
send/4, 3164, 3237
send/5, 3161
send_binary/2, 1536
send_comment/2, 1516
send_data/2, 1514
send_event/3, 1515
send_heartbeat/1, 19, 3241
send_id_data/3, 1515
send_json/2, 1517, 1537
send_pingreq/1, 2314
send_retry/2, 1516
send_term/2, 1518, 1538
send_text/2, 1535
sentence_splitter_protocol, 3508
sentence_type/2, 2447
sequence/2, 824, 2573, 3093, 3765
sequence/3, 3600
sequence/4, 2810, 3598, 3601
sequence/5, 3598
sequence_count/2, 280

sequence_dataset_protocol, 3092
sequence_flags/2, 279, 284
sequence_grammars, 1057
sequence_number/2, 229
sequential_occurrences/2, 3613
sequential_occurrences/3, 3613
sequential_pattern_mining_common, 3094
serialize/2, 84, 2297, 2774, 3078, 3146, 3191
serialize/4, 84, 2297, 2775, 3078, 3146, 3191
series_length/1, 3494
series_observation_summary/4, 3462
serve/3, 1453, 2793
serve/4, 1361, 1442, 1527
serve/5, 1361, 1443, 1527
serve/6, 1444
serve_connection/3, 1454
serve_listener/4, 1502
serve_listener/5, 1503
serve_once/3, 1440, 1500
serve_once/5, 1521, 1539
serve_once/6, 1521, 1540, 1564
serve_once/7, 1564
serve_sse/4, 1456
serve_sse_once/5, 1501
serve_until_shutdown/4, 1504
serve_until_shutdown/5, 1445, 1504, 1566
serve_until_shutdown/6, 1446, 1567
serve_websocket/4, 1455
serve_websocket_once/5, 1441, 1501
server_accept/4, 3180
server_accept/5, 3179
server_close/1, 3180
server_control_/3, 1451
server_control_counter_/1, 1451
server_open/2, 3179
server_open/3, 3179
server_open/4, 3178
server_options_/1, 2212, 2233
server_property/2, 1448
server_running_/1, 2082
server_shutdown_/0, 2082
server_socket_/1, 2081
servers/1, 1428, 2523
session/2, 1581
session_count/2, 1581
session_seed_/1, 1343, 1465, 3058
session_state_/2, 1343, 1465, 3058
sessions/2, 1581
set, 3098
set(Type), 3099
set/1, 3788
set/3, 1481, 2248
set/4, 2810, 2948
set/5, 2247
set_binary_input/1, 2023
set_binary_input/2, 2022
set_binary_input/3, 2022
set_binary_output/1, 2031
set_binary_output/2, 2030
set_binary_output/3, 2030
set_element/2, 1754
set_field/2, 1744
set_flag_value/2, 908
set_flag_value/3, 908
set_matrix_entry/5, 2835
set_monitor/4, 814
set_rating_scale/3, 176
set_seed/1, 63, 2801
set_spy_point/4, 819
set_text_input/1, 2019
set_text_input/2, 2019
set_text_input/3, 2018
set_text_output/1, 2026
set_text_output/2, 2025
set_text_output/3, 2025
set_write_max_depth/1, 628
setp, 3101
setrange/5, 2949
setup/0, 2016, 2651
sgd_classifier, 3111
sha1, 1166
sha224, 1167
sha256, 1168
sha256sum_command/1, 2658
sha384, 1169
sha3_224, 1171
sha3_256, 1172
sha3_384, 1173
sha3_512, 1174
sha512, 1175
sha512_256, 1176
shake128(OutputBytes), 1177
shake256(OutputBytes), 1178
shell, 3803
shell(Interpreters), 3805
shell/1, 2597
shell/2, 2597
shell_command/1, 799
shell_expansion(Mode), 3806
shift_matrix_diagonal/3, 2104
shl64/3, 1151
short_flags/1, 449
shr64/3, 1152
shrink/3, 62
shrink_sequence/3, 62
shrinker/1, 62
shutdown_control_/1, 2234
shutdown_control_/3, 1569
shutdown_requested_/2, 1569
shutdown_seed_/1, 1568
shutdown_server/1, 2071
sign//1, 1056

sign/4, 2702
sign/5, 1818, 1845
sign_payload/5, 1819, 1845
sign_request/8, 3060
signaling_field/2, 214
signed_integer_to_bytes/4, 195
signed_integer_to_bytes/5, 195
signing_input/4, 1827
similarity/3, 2937
similarity_metric_common, 2931
similarity_metric_protocol, 2936
simplex, 2135
simulated_annealing(Problem), 3127
simulated_annealing(Problem,Random), 3129
simulated_annealing_protocol, 3132
single_byte_character_set(MaxCode), 331
single_classification_ripple_down_rules, 3022
singleton/2, 113, 1676
singletons/2, 3654
sink_closest_min_cut/5, 944
siphash_2_4, 1179
siphash_2_4(Key), 1180
sismember/4, 2960
size/2, 752, 1190, 2186, 3020, 3105, 3552
size/3, 3198
size_metric, 432
sk_language_profile, 1939
skewness/2, 3223
skipped_/1, 2041
skipping_/0, 612
skipping_unleashed_/1, 613
sl_language_profile, 1940
sleep/1, 2610
sliding_window/3, 3346
slope_one_recommender, 3136
smembers/3, 2960
smt, 3137
smt_expansion, 3138
smt_output, 3139
smt_problem_protocol, 3141
smt_process, 3142
smt_program, 3145
smt_protocol, 3147
smt_statement/2, 3142
smtp, 3157
smtp_process_transport, 3158
smtp_protocol, 3160
snapshot_/6, 566
snowflakeid, 3166
snowflakeid(Representation,EpochMilliseconds,DatacenterIdBits,WorkerIdBits,TimestampBits,NodeBits,Sequence), 3167
snowflakeid_instagram, 3169
snowflakeid_instagram(Representation), 3170
snowflakeid_protocol, 3171
snowflakeid_sonyflake, 3172
snowflakeid_sonyflake(Representation), 3173
snowflakeid_twitter, 3174
snowflakeid_twitter(Representation), 3175
so_language_profile, 1941
socket, 3176
softmax/2, 3643
softmax/3, 3643
software_heritage_identifier/1, 57
solution_cost/2, 45
solve/2, 87, 2128, 2299, 2491, 2777, 3080, 3149, 3193
solve/3, 88, 2129, 2300, 2491, 2777, 3081, 3150, 3194
solve/4, 2468
solve/5, 2468
solve/6, 2469
solve/8, 473
solve_cholesky/3, 2118
solve_linear_system/3, 2106
solve_linear_system/4, 2987
solve_linear_systems/3, 2107
solve_lower_triangular_matrix/3, 2119
solve_upper_triangular/3, 2119
solve_upper_triangular_matrix/3, 2119
sonar_report, 2046
sort/2, 3100, 3619
sort/3, 3619
sort/4, 3620
sort_by_decreasing_score/2, 882
sort_item_supports/2, 3097
sort_patterns/2, 962, 3096
sorted_median/3, 3213
sorted_observations/5, 3475
souffle, 3182
souffle_expansion, 3183
souffle_output, 3184
souffle_problem_protocol, 3186
souffle_process, 3187
souffle_program, 3190
souffle_protocol, 3192
souffle_statement/2, 3187
soundex/2, 3319
soundex_match/2, 3320
source_bits/2, 1606
source_file_extension/1, 730
space//0, 1046
space_atom/2, 2548
spacecraft_id/2, 219, 223
spaces//0, 1047
spade_pattern_miner, 3200
spdx_license_schema_/1, 3090
spearman_similarity, 2937
spearman_similarity/3, 2938
spec/1, 2226, 2236
spec_version/1, 2502
spectral_clusterer, 3201
spelling_normalizer_protocol, 3378
splaytree, 754

split/3, 3579
split/4, 3620
split_aos_packet_zone/3, 254
split_packet_zone/4, 252
split_pairs/3, 873, 2934
split_sentence_codes/3, 3513
split_sentences/2, 3509
split_sentences/3, 3509
split_sorted_rows/5, 390
split_string/4, 3327
spy/1, 623
spy/3, 624
spy/4, 625
spy_point/4, 819
spy_point_/4, 817
spying/1, 624
spying/3, 625
spying/4, 626
sqids, 3203
sqids(Representation), 3204
sqp_active_set(Problem), 474
squared_deviations/3, 872
squared_error_sum/6, 3214
squared_mahalanobis_distance/4, 2350
squared_mahalanobis_distance/5, 2350
squared_norm/2, 1908
squares_and_cubes/6, 3211
squares_and_hypers/6, 3212
squeeze/3, 1140
srem/4, 2960
sse_headers/1, 2232
sse_mode_/1, 2235
sse_output_/1, 2235
st_language_profile, 1942
stabilize_vector_sign/2, 2115
stabilize_vector_sign/3, 2116
standard_cauchy/3, 2824
standard_deviation/2, 3222
standard_error/2, 3228
standard_exponential/1, 2824
standard_gamma/2, 2824
standard_normal/1, 2825, 3715
standard_normal_density/2, 3717
standard_normal_distribution/2, 3717
standard_normal_log_density/2, 3717
standard_normal_quantile/2, 3718
standard_normal_samples/2, 3715
standard_t/2, 2823, 3721
standard_t_density/3, 3722
standard_t_distribution/3, 3723
standard_t_log_density/3, 3723
standard_t_quantile/3, 3724
standard_t_samples/3, 3722
start/0, 2759, 3804
start/2, 2203
start/3, 2203
start/4, 1448, 2203, 2237
start/5, 1449, 2203
start_redirect_to_file/2, 630
started_by/2, 1671
starts/2, 1670
state_energy/2, 974, 3134, 3362
state_key/3, 3357
statistics, 3210
statistics/2, 2131
statisticsp, 3216
stats/1, 26
stats/2, 2255
stats/3, 2256
status/1, 1325
status/2, 1302, 2130, 2492, 3153
status_file_/1, 2409
stem/2, 3233
stemmer_protocol, 3232
stems/2, 3233
stirling_first/3, 3628
stirling_second/3, 3629
stn, 3112
stn_protocol, 3114
stomp, 3234
stop/0, 2759
stop/1, 1450
stop_condition/3, 46, 775, 975, 2157, 2682, 3135, 3362
stop_redirect_to_file/0, 630
stop_word/1, 1924, 3245, 3246
stop_words(Representation,Language), 3242
stop_words_language_protocol, 3244
stop_words_protocol, 3245
stopword_language_detector, 1943
stopwords_af, 3247
stopwords_ar, 3248
stopwords_bg, 3249
stopwords_bn, 3251
stopwords_br, 3252
stopwords_ca, 3253
stopwords_cs, 3254
stopwords_da, 3255
stopwords_de, 3256
stopwords_el, 3258
stopwords_en, 3259
stopwords_eo, 3260
stopwords_es, 3261
stopwords_et, 3262
stopwords_eu, 3263
stopwords_fa, 3265
stopwords_fi, 3266
stopwords_fr, 3267
stopwords_ga, 3268
stopwords_gl, 3269
stopwords_ha, 3270
stopwords_he, 3272
stopwords_hi, 3273

stopwords_hr, 3274
stopwords_hu, 3275
stopwords_hy, 3276
stopwords_id, 3277
stopwords_it, 3279
stopwords_ja, 3280
stopwords_ko, 3281
stopwords_ku, 3282
stopwords_la, 3283
stopwords_lt, 3284
stopwords_lv, 3286
stopwords_mr, 3287
stopwords_ms, 3288
stopwords_nl, 3289
stopwords_no, 3290
stopwords_pl, 3291
stopwords_pt, 3293
stopwords_ro, 3294
stopwords_ru, 3295
stopwords_sk, 3296
stopwords_sl, 3297
stopwords_so, 3298
stopwords_st, 3300
stopwords_sv, 3301
stopwords_sw, 3302
stopwords_th, 3303
stopwords_tl, 3304
stopwords_tr, 3305
stopwords_uk, 3307
stopwords_ur, 3308
stopwords_vi, 3309
stopwords_yo, 3310
stopwords_zh, 3311
stopwords_zu, 3312
store_regressor_options/3, 1960
store_set_cookies/3, 1469
stored_session_/6, 1485
strata/1, 577
strata_from_numbers/2, 569
stream_position/1, 2038
stream_to_bytes/2, 2895
stream_to_bytes/3, 2896
stream_to_chars/2, 2894
stream_to_chars/3, 2894
stream_to_codes/2, 2893
stream_to_codes/3, 2894
stream_to_terms/2, 2895
stream_to_terms/3, 2895
streams/3, 3159
streamvars, 2065
strength_dictionary/2, 2840, 2851
strength_pairs/3, 2840, 2849
string(Representation), 3323
string_chars/2, 3325
string_codes/2, 3325
string_concat/3, 3325
string_distance(Representation), 3314
string_length/2, 3326
string_lower/2, 3327
string_upper/2, 3326
strings/2, 3547
strings/3, 3547
strlen/3, 2949
strongest_paths/2, 3092
strongly_connected/3, 2841, 2850
strongly_connected_components/2, 1065
strongly_connected_components/4, 2841, 2851
sub_diagram_/1, 705, 711, 721, 727
sub_diagram_/2, 691, 697
sub_directory/2, 406
sub_library/2, 407
sub_problem(Problem,OuterIteration,StartPoint,EqLar
476
sub_string/5, 3326
subclass/1, 1210
subclasses/1, 1211
subdivision/4, 1701, 1702
subfactorial/2, 3628
subject, 632
sublist/2, 3620, 3669
submilliseconds/2, 302
subprocess_coverage_hook, 2405
subprocess_mutation_hook, 2407
subscribe/4, 2312, 3237
subscription_/3, 2216
subscription_/4, 2234
subsequence/2, 3334
subsequence/3, 3335, 3621
subsequence/4, 3621
subsequence_at_indices/3, 3343
subsequence_length/2, 3350
subsequences, 3330
subsequences/2, 3334
subsequences/3, 3334
subsequences_protocol, 3332
subsequences_with_min_span/3, 3347
subset/2, 113, 1677, 3108
subslices/2, 3346
substitute/4, 3621
subsumes/2, 1665, 3652
subterm/2, 3652, 3825
subtract/3, 3108, 3622, 3669
subtract_duration/3, 583
subtract_matrices/3, 2098
subtract_vectors/3, 2092
subunit_v1_output, 2047
subunit_v1_report, 2048
subunit_v2_output, 2049
subunit_v2_report, 2050
succ/2, 3600
successive_shortest_paths, 953
successive_shortest_paths(Dictionary), 954

sufficient/2, 1406
suffix/2, 3576, 3622, 3670
suffix/3, 3622
suffix_alias/2, 2287
suffix_data/2, 252
sum/2, 3217, 3637
sum_of_squared_deviations/3, 872
sum_of_squares/2, 3227
sum_of_squares/4, 3213
sum_of_squares_list/2, 873, 2935
summary/1, 2142
summary_/2, 1995
superclass/1, 1211
superclasses/1, 1211
supervised_dimension_reduction_dataset_protocol, 794
supplier/1, 52
support_count_/2, 565
support_edge_/3, 566
supported_algorithm/1, 1837
supported_archive/1, 2659
supported_editor_url_scheme_prefix/1, 677
supported_method/1, 1306
supported_range/2, 3447
supported_request_scheme/1, 1506
supported_specs/1, 2206
supported_url_archive/1, 2659
supported_websocket_scheme/1, 1506
suppress_binary_output/0, 2018
suppress_goal_hook, 1239
suppress_text_output/0, 2018
suspend_monitor/0, 818
sv_language_profile, 1944
svr_regression, 3350
sw_language_profile, 1945
swap/1, 837, 3777
swap/2, 2808
swap_consecutive/2, 2808
symbol_codes/2, 85
symdiff/3, 3108
symmetric_closure/2, 1062
symmetric_eigen/3, 2110
symmetric_eigen/4, 2110
symmetric_eigen/5, 2111
symmetric_eigen_partial/7, 2112
symmetric_key_bytes/2, 1839
symmetrical_uncertainty_score, 902
symmetrical_uncertainty_score(Discretization), 903
synchronization_flag/2, 233

T

t/4, 3724
t_density/5, 3726
t_distribution/5, 3727
t_log_density/5, 3727
t_quantile/5, 3728
t_samples/5, 3725
tab//0, 1047
tabs//0, 1047
tabu_key/2, 3360
tabu_search(Problem), 3352
tabu_search(Problem,Random), 3353
tabu_search_problem_protocol, 3358
tabu_tenure/4, 3361
tai_minus_utc_for_tai_unix/2, 3444
tail/2, 3338
tail1/2, 3339
tails/2, 3337
tails1/2, 3339
take/3, 3623
take/4, 3624
take_first_k/3, 379
take_pending/3, 1583
talker/2, 2447
tap_output, 2052
tap_report, 2054
tar_command/1, 2658
target/1, 797, 1322, 2964
target/2, 1301, 1366
target_predicate/3, 2393
target_predicate_clause_index/4, 2393
target_predicate_directive/3, 2394
target_predicate_directive_index/4, 2395
target_reached/3, 2161
target_scope_directive/3, 2394
target_scope_directive_index/4, 2394
target_supervised_dimension_reduction_dataset_proto 796
target_uses_directive/3, 2395
target_uses_directive_index/4, 2395
targets/2, 881
tcb_minus_tdb_approx/3, 3443
tcg_minus_tt_approx/3, 3443
tdb_minus_tt_approx/3, 3443
team/3, 2843
team_member/4, 2844
temporal_pairwise_dataset_connected_components/2, 2870
temporal_pairwise_dataset_declared_items/2, 2868
temporal_pairwise_dataset_games/2, 2869
temporal_pairwise_dataset_games/3, 2870
temporal_pairwise_dataset_items/2, 2869
temporal_pairwise_dataset_periods/2, 2869
temporal_pairwise_dataset_summary/2, 2870
temporal_pairwise_ranking_dataset_protocol, 2888
temporary_directory/1, 2601
temporary_file_/1, 3364
temporary_tls_credentials/3, 1420
temporary_tls_credentials_files/3, 1420

term, 3649
term/2, 3067
term/4, 3067
term_expansion/2, 481
term_io, 3363
term_io_protocol, 3365
termp, 3650
terms_to_array/2, 1751
test/1, 2007
test/2, 2039
test/3, 2039
test_/2, 2040
test_count_/1, 2053, 2055
test_time_/3, 2039
text_body/2, 3005
text_case_folding, 3379
text_diacritics, 3381
text_entities, 3383
text_file_assertion/3, 2036
text_file_assertion/4, 2035
text_input_assertion/2, 2021
text_input_assertion/3, 2021
text_normalization_profile_protocol, 3384
text_normalizer(Representation,Profile), 3386
text_normalizer_protocol, 3387
text_output_assertion/2, 2028
text_output_assertion/3, 2028
text_output_assertion/4, 2027
text_output_contents/1, 2029
text_output_contents/2, 2029
text_output_contents/3, 2029
text_unicode, 3400
text_vectorizer, 3404
text_vectorizer_protocol, 3405
text_whitespace, 3401
tfidf_recommender, 3413
th_language_profile, 1946
theta_correction/3, 3432
theta_forecasting, 3425
thurstone_mosteller_ranker, 3435
tighten_variable_bounds/5, 2123
time, 589
time_expanded_network, 955
time_expanded_network(Dictionary), 956
time_of_day/4, 300
time_points/2, 3116
time_scales, 3436
time_scales_data, 3437
time_scales_protocol, 3446
time_series_dataset_protocol, 3493
time_series_regression, 3495
time_stamp/1, 2607
time_string/3, 1688
time_type/2, 3679
time_type/3, 3679
time_type/4, 3678
timeout, 3498
timeout/1, 2277
timep, 590
timestamp/2, 3702
timestamp/8, 3703
timestamp_/6, 1994, 1997
title/3, 1722
title_case/2, 3393
tl_language_profile, 1947
tle_orbits, 3500
tle_orbits_protocol, 3501
to_deviation/2, 2838
to_expected/1, 3780
to_expected/2, 2585
to_list/2, 114
to_mu/2, 2837
to_optional/1, 839, 3780
to_phi/2, 2837
to_rating/2, 2837
to_uuid/2, 3575
today/3, 580
token_hex/2, 522
token_info/2, 1388
token_urlsafe/2, 522
tokenize/2, 3515
tokenize/3, 3515
tokenize_codes/3, 3513
tokenize_sentences/2, 3516
tokenize_sentences/3, 3517
tokenizer(Representation,Language), 3511
tokenizer_language_protocol, 3512
tokenizer_protocol, 3514
tokenizer_rules, 3518
tolerance_equal/4, 2014, 3632
toml, 3521
toml(ObjectRepresentation,PairRepresentation,String 3523
toml(StringRepresentation), 3522
toml_protocol, 3524
tool/1, 1201
tool_call/3, 2241
tool_call/4, 2242
tool_call_round/4, 2196
tool_descriptors_to_json/3, 2219
tool_diagnostics_common, 3526
tool_diagnostics_protocol, 3527
tool_input_schema/4, 2222
tool_output_schema/4, 2222
tool_ui/2, 2244
tools/0, 1200
tools/1, 2241
toon, 3533
toon(ObjectRepresentation,PairRepresentation,String 3535
toon(StringRepresentation), 3534

toon_protocol, 3536
top/3, 1191
top_k/3, 2921
top_next/5, 1191
topological_sort/2, 1063
topological_sort/3, 1087
totp/5, 2613
totp_verify/7, 2614
touch/3, 2253
toychrdb, 3538
tr_language_profile, 1948
trace/0, 622
trace_event/2, 488
tracing_/0, 612
train_linear_model/6, 2981
train_ridge_model/6, 2982
train_with_cleanup/1, 2
training_totals/6, 3431
transaction/2, 965
transaction_dataset_protocol, 964
transform/3, 789, 3407
transform/4, 513
transform_all/3, 3408
transitive_closure/2, 1062
transitive_reduction/2, 1088
transpose/2, 1062, 3647
transpose_matrix/2, 2099
traverse/3, 825, 2573, 3765
treap_set, 3110
tree/2, 938
triangular/4, 2821
triangularize/2, 2987
trie(Representation), 3545
trigger/2, 1366
trigger_name/2, 1366
triggered_breakpoint_/4, 617
triggered_breakpoint_enabled_/2, 617
trigram_counts/1, 1921, 1924
trigram_counts/2, 1921
trigram_counts_/2, 1921
trigrams/2, 2428
trim/2, 3328
trim/3, 3328
trim_left/2, 3329
trim_left/3, 3329
trim_ows_codes/2, 1330
trim_right/2, 3329
trim_right/3, 3330
trim_trailing_ows_codes/2, 1330
trimmed_mean/3, 3227
true/1, 1748
trueskill_ranker, 3552
truncated_svd_projection, 3554
trust_region_newton_cg(Problem), 2165
truth_literal_flip(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation), 2409
try_add_constraints/4, 3120
try_tool_call_3/7, 2221
ts_findall_in_noblock/3, 2086
ts_findall_rd_noblock/3, 2086
ts_in/4, 2084
ts_in_list/4, 2085
ts_in_noblock/2, 2084
ts_out/1, 2084
ts_rd/4, 2085
ts_rd_list/4, 2086
ts_rd_noblock/2, 2085
tsne_projection, 3556
tsv, 3557
tsv(Header), 3558
tsv(Header,Comments), 3559
tsv_protocol, 3560
tt_minus_tai/2, 3442
ttl/3, 2952
tuple_/1, 2083
tutor, 3568
tutor_explanations, 3569
two3tree, 755
tx_commit/1, 18
tx_rollback/1, 19
tx_select/1, 18
type, 3656
type/1, 449, 3659
type/2, 278, 1555
type/3, 2278, 2954
type_entity_/4, 1979
typeid, 3571
typeid(Representation), 3572
typeid_protocol, 3573
tzdb_version/1, 3697
tzif, 3671
tzif_protocol, 3673
tzif_zone_ids, 3696

U

uk_language_profile, 1949
ulid, 3698
ulid(Representation), 3699
ulid_protocol, 3700
ulid_types, 3704
umap_projection, 3705
unauthorized_response/3, 1253, 1351, 1389
unauthorized_response/4, 1254, 1352, 1390
undefined/2, 331
undirected_graph_common, 1080
unexpected/1, 832
unexpecteds/2, 824
uniform/1, 2821
uniform/3, 2821
uninstall/0, 2641
uninstall/1, 2640
uninstall/2, 2640

union/3, 113, 757, 1087, 1677, 3109
union/4, 3109, 3709
union_all/3, 3710
union_find, 3707
union_find_protocol, 3708
union_impl/3, 757
union_impl_list/3, 758
unique_min_cut/4, 944
univariate_distributions(Random), 3712
univariate_distributions_protocol, 3713
univariate_function_protocol, 2485
universal/1, 1675
unix_seconds/2, 307
unix_to_date_time/2, 582
unknown_predicate_called_/2, 3819
unpin/0, 2653
unpin/1, 2653
unregister/2, 1580
unsafe_set_flag_value/2, 911
unsat_core_name/2, 2493, 3154
unsubscribe/3, 3238
unsubscribe/4, 2313
unweighted_directed_graph, 1084
unweighted_directed_graph(Dictionary), 1086
unweighted_graph_common(Dictionary), 1088
unweighted_graph_protocol, 1090
unweighted_undirected_graph, 1092
unweighted_undirected_graph(Dictionary), 1093
unzip/2, 3837
update/0, 799, 2639, 2668
update/1, 632, 2639, 2667
update/2, 2638, 2667
update/3, 143, 575, 746, 841, 1641, 1864, 2637, 3497
update/4, 745, 3549
update/5, 746, 2188, 3550
update_fecf/2, 214, 218, 225, 230, 235
update_hash_state/3, 1158
update_impl/3, 766
update_in/4, 2441
update_in/5, 2442
update_item_parameters/10, 2837
update_penalty/6, 464
update_ratings/3, 172, 1733, 3416
update_score/4, 2887
update_smoothing/7, 845
update_strengths/4, 2839, 2848
update_target_predicate_clause_index_/2, 2396
update_target_predicate_directive_index_/2, 2398
update_target_scope_directive_index_/2, 2397
update_target_uses_directive_index_/2, 2398
updated_observation_diagnostics/3, 3466
updated_profiles/6, 182
upn_metric, 433
upper_case/2, 3392
upper_triangular_part/2, 2105
upper_triangular_part/3, 2105
uppercase_ascii_codes/2, 1330
ur_language_profile, 1950
uri_template(Representation), 3748
url(Representation), 3750
url_scheme/2, 2548
us_ascii, 332
usable_examples/3, 1957
user, 498
user_data/2, 280
user_mean_rating/3, 2918
user_profile/5, 177
user_vector/3, 2917
userinfo/4, 2530, 2539
users/2, 2917
uses_diagram, 732
uses_diagram(Format), 733
uses_directive_resource_deletion(Entity,Predicate,I 2410
utc_date_time_to_instant/2, 3451
utc_to_local/3, 584
utc_to_local_tz/3, 594
utf_16_character_set(Endian), 333
utf_16be, 334
utf_16le, 335
utf_32_character_set(Endian), 336
utf_32be, 337
utf_32le, 338
utf_8, 339
utf_8_character_set, 340
utm_crs/2, 513
utm_zone/2, 513
uuid, 3755
uuid(Representation), 3756
uuid_max/1, 3761
uuid_nil/1, 3761
uuid_null/1, 3760
uuid_protocol, 3757
uuid_v1/2, 3758
uuid_v1/3, 3758
uuid_v3/3, 3759
uuid_v4/1, 3759
uuid_v5/3, 3759
uuid_v7/1, 3760
uuid_v7/2, 3760

V

valid/0, 448
valid/1, 218, 222, 251, 306, 309, 1617, 1668, 3229, 3577, 3653, 3670, 3749, 3751, 3773
valid/2, 3659
valid/3, 582, 591
valid_anomaly_detector/1, 36

valid_association_rule_miner/1, 108
valid_attribute_declarations/1, 2971
valid_attribute_names/1, 351, 371, 2971
valid_baseline_class_values/1, 33
valid_bearer_token/1, 1395
valid_bits/1, 1608
valid_channel_reassembly_state/1, 249
valid_class_values/1, 352
valid_classifier/1, 360
valid_classifier_metadata/2, 357
valid_classifier_metadata/3, 357
valid_clusterer/1, 382
valid_clusterer_metadata/3, 372
valid_column_type/1, 3192
valid_continuous_encoders/1, 371
valid_conversion/3, 3452
valid_cookie/1, 1280
valid_coordinate/1, 1008
valid_coordinate/2, 511
valid_crs/1, 509
valid_cut/5, 929
valid_date/3, 1690
valid_date_time/1, 587
valid_descriptor/1, 312
valid_diagnostic_choice/3, 373
valid_diagnostic_count/3, 373, 2972
valid_diagnostics/4, 1650
valid_dimension_reducer/1, 790
valid_dimension_reducer_metadata/1, 787
valid_discontinuity_policy/1, 240, 250, 288
valid_discrete_encoders/1, 372
valid_discrete_values/1, 352, 2971
valid_encoded_rows/2, 2973
valid_error_metrics/5, 1651
valid_feature_discretization/1, 864
valid_feature_labels/1, 2975
valid_feature_types/2, 352
valid_fitting/2, 3434
valid_flow/3, 929
valid_forecaster/1, 3490
valid_forecaster_metadata/2, 3465
valid_forecaster_metadata/3, 3465
valid_geohash/1, 987
valid_identifier/1, 85, 2298, 3192
valid_instant/1, 3451
valid_item_value_pairs/2, 2857
valid_itemset_patterns/2, 963
valid_label_code/1, 1726
valid_labels/1, 1726
valid_linear_encoders/1, 353, 786
valid_linear_model_convergence/1, 2979
valid_linear_model_diagnostics/1, 2973
valid_map_dispatch/1, 285
valid_min_cost_flow/4, 930
valid_min_cost_flow/5, 930
valid_mixed_encoders/1, 372
valid_mixed_vectors/2, 372
valid_model/1, 1960, 3020, 3433
valid_option/1, 2588
valid_options/1, 2589
valid_pattern_miner/1, 2735
valid_pattern_miner_metadata/5, 2731
valid_prefix/1, 3577
valid_profile/1, 237
valid_projection_components/2, 787
valid_ranker/1, 2861
valid_ranker_metadata/2, 2857
valid_reassembled_segment/1, 286
valid_reassembly_state/1, 240, 249, 287
valid_recommender/1, 2928
valid_recommender_metadata/2, 2921
valid_recommender_metadata/3, 2922
valid_regression_encoders/1, 2971
valid_regression_tree/2, 2976
valid_regressor/1, 2993
valid_regressor_metadata/2, 2972
valid_regressor_options/1, 2972
valid_regularization_grid/1, 1961
valid_residual_history/5, 3464
valid_residual_positions/3, 3477
valid_result/1, 2769
valid_scale/1, 3447
valid_score/1, 2885
valid_score_ranker_diagnostics/3, 2886
valid_seasonality/4, 3434
valid_segment/1, 284
valid_selector/1, 888
valid_selector_metadata/2, 883
valid_selector_metadata/3, 883
valid_sequence_patterns/2, 3097
valid_set_cookie/1, 1281
valid_st_flow/5, 930
valid_suffix/1, 3577
valid_unicode_scalar/1, 315
valid_until_date/1, 53
valid_vectorizer/1, 3411
validate/1, 914, 1002, 1040, 1841, 2716, 3810
validate/2, 1002, 1040, 1810, 3810
validate/3, 912, 1811
validate_claim/3, 1821, 1823, 2706, 2709
validate_claims/3, 1820, 1822, 2705, 2709
validate_constraint_data/4, 465
validate_document/1, 2513
validate_document/2, 2514
validate_document_root/1, 1305
validate_grouped_dataset/1, 2876
validate_grouped_dataset/2, 2878
validate_header/1, 1836
validate_http_request/3, 2515
validate_http_request/4, 2515
validate_http_response/3, 2516
validate_http_response/4, 2517

validate_key_set/1, 2707
validate_multiplayer_dataset/1, 2877
validate_multiplayer_dataset/2, 2877
validate_non_negative_integer_option/4, 1574
validate_numeric_vector/2, 465
validate_pairwise_dataset/1, 2875
validate_pairwise_dataset/2, 2875
validate_pairwise_measurement_dataset/1, 2875
validate_pairwise_measurement_dataset/2, 2876
validate_relative_path/1, 1305
validate_request/1, 1305
validate_request/3, 2514
validate_request/4, 2515
validate_response/3, 2516
validate_response/4, 2516
validate_role/0, 1594
validate_temporal_pairwise_dataset/1, 2876
validate_temporal_pairwise_dataset/2, 2877
validate_terminals/3, 932
validate_type/1, 913
validated, 3762
validation, 3766
validation(Validation), 3771
valids/2, 3763
value/1, 3788
value/3, 3647
value_reference/2, 1748
values/2, 751, 3548, 3646
values/3, 3548
variable/4, 2125
variable/6, 2126
variable_value/3, 2131
variables/2, 3079, 3654, 3749
variance/2, 3224
variance/6, 3212
variance_score, 904
variance_values/2, 870
variant/2, 2012, 3653
varlist, 3661
varlistp, 3662
varnumbers/2, 3655
varnumbers/3, 3655
vector_norm/2, 2932
vector_norm/3, 2093
vector_value/3, 2100
vectorizer_options/2, 3412
verified_accounts/2, 1114
verify/3, 1256, 1407
verify/4, 1816, 2703
verify/5, 1817, 1844, 1847, 2564, 2704
verify_commands_availability/0, 2651
verify_fecf/1, 215, 219, 225, 230, 235
verify_id_token/4, 2532, 2556
verify_id_token/5, 2534, 2560
verify_password_hash/2, 532
verify_signature/5, 1818
version/1, 49, 1322, 1325
version/2, 219, 223, 278, 1302, 2255
version/6, 2618
version_persistent_by_default/1, 1315
versions/3, 2627
vertex_neighbors_to_edges/4, 1067
vertices/2, 926, 1070
vi_language_profile, 1951
vincenty_distance/3, 1009
virtual_channel_frame_count/2, 214, 233
virtual_channel_id/2, 220, 223
vocabulary/2, 3411
void/1, 1749
void_element/1, 1245
von_mises/3, 2822

W

wait/2, 2782
wait/3, 2782
waiting_/3, 2083
wald/3, 2816
wall_time/1, 2608
warning/1, 2141
warning_sequence_/1, 2143
warnings/1, 2142
weakly_connected_components/2, 1065
websocket_accept/2, 1550
websocket_opening_key/1, 1550
week_of_year_iso/2, 586
weekday/2, 586
weibull/3, 2820
weight_variances/2, 144
weighted_directed_graph, 1095
weighted_directed_graph(Dictionary), 1096
weighted_graph_common(Dictionary), 1097
weighted_graph_protocol, 1100
weighted_mean/3, 3220
weighted_undirected_graph, 1103
weighted_undirected_graph(Dictionary), 1104
welcome/0, 3804
well_known_url/2, 1402
wfind/3, 1099
when/2, 501
whiledo/2, 2179
white_space//0, 1046
white_spaces//0, 1046
why/3, 126
window_interval/3, 3125
window_relation/4, 3126
windows_1250, 341
windows_1251, 342
windows_1252, 343
windows_1253, 344

windows_1254, 345
windows_1257, 346
winsert_neighbor/4, 1098
with_connection/1, 25
with_output_to/2, 3376
within_distance/4, 1014
without//2, 1059
wkt_wkb, 3807
wkt_wkb_protocol, 3808
wmc_metric, 434
wneighbors/3, 1102
word32_hex/2, 1144
word64_hex/2, 1149
working_directory/1, 2601
working_tree_clean/1, 1031
working_tree_status/2, 1031
wpairs_to_edges/2, 1067
wrapper, 3812
wremove_neighbor/4, 1098
wremove_vertex_from_all/3, 1099
write_file/3, 555, 3566
write_frame/2, 1547
write_framed_message/2, 1805
write_max_depth/1, 628
write_max_depth_/1, 615
write_message/2, 1555, 1591, 1804
write_message/3, 1592
write_message/4, 1593
write_message/5, 1593
write_packet/2, 2317
write_request/2, 1276
write_response/2, 1453
write_stream/3, 556, 3567
write_term_to_atom/3, 3371
write_term_to_chars/3, 3372
write_term_to_chars/4, 3372
write_term_to_codes/3, 3373
write_term_to_codes/4, 3374
write_to_atom/2, 3371
write_to_chars/2, 3373
write_to_codes/2, 3375
write_to_file_hook(File), 1240
write_to_file_hook(File,Options), 1241
write_to_stream_hook(Stream), 1242
write_to_stream_hook(Stream,Options), 1243
wvertex_neighbors_to_edges/4, 1068

X

x25519, 537
x25519_keypair/2, 538
x25519_public_key/2, 539
x25519_shared_secret/3, 539
xchacha20/4, 541
xchacha20_poly1305, 540
xchacha20_poly1305_decrypt/5, 544
xchacha20_poly1305_encrypt/5, 543
xchacha20_subkey_and_nonce/4, 542
xhtml11, 1248
xml, 3823
xml_to_document/3, 3826
xml_to_gpx/2, 1041
xml_whatwg_entities, 3402
xml_whatwg_named_entity/2, 3403
xor64/3, 1150
xor_bytes/3, 153, 534
xref_diagram, 734
xref_diagram(Format), 736
xunit_net_v2_output, 2055
xunit_net_v2_report, 2057
xunit_output, 2058
xunit_report, 2059

Y

yaml, 3829
yaml_protocol, 3830
year/2, 299
yo_language_profile, 1952

Z

z_normalization/2, 3224
z_score_anomaly_detector, 3833
zadd/5, 2961
zap_to_port_/1, 614
zcard/3, 2962
zero_bits/2, 1608
zero_from_sign/2, 1609
zero_matrix/2, 2834
zero_or_more//0, 1059
zero_or_more//1, 1058
zero_or_more//2, 1058
zero_vector_from_values/3, 2981
zero_vector_like/2, 783
zh_language_profile, 1953
zip/2, 3836
zip/3, 838, 2584, 3779, 3836
zip_at_index/4, 3845
zipperp, 3835
zlist, 3844
zone/3, 3677
zone_id_kind/2, 3697
zones/1, 3678
zones/2, 3678
zrange/5, 2962
zrank/4, 2962
zrem/4, 2961
zscore/4, 2963
zu_language_profile, 1954