
Logtalk APIs

Release v3.99.0

Paulo Moura

Apr 09, 2026

CONTENTS

1	Libraries	1
1.1	ada_boost	1
1.1.1	ada_boost	1
1.2	amqp	4
1.2.1	amqp	4
1.2.2	amqp_pool	24
1.3	application	29
1.3.1	application_common	30
1.3.2	application_protocol	31
1.4	arbitrary	42
1.4.1	arbitrary	42
1.5	assertions	48
1.5.1	assertions	49
1.5.2	assertions(Mode)	50
1.5.3	assertions_messages	52
1.6	assignvars	53
1.6.1	assignvars	53
1.6.2	assignvarsp	54
1.7	avro	58
1.7.1	avro	58
1.8	base32	61
1.8.1	base32	61
1.9	base58	63
1.9.1	base58	63
1.10	base64	65
1.10.1	base64	65
1.10.2	base64url	67
1.11	base85	69
1.11.1	base85	69
1.12	c45	71
1.12.1	c45	71
1.13	cbor	74
1.13.1	cbor	74
1.13.2	cbor(StringRepresentation)	75
1.14	ccsds	77
1.14.1	ccsds	77
1.14.2	ccsds(SecondaryHeaderLength)	78
1.14.3	ccsds_types	85
1.15	character_sets	86
1.15.1	character_set	86

1.15.2	character_set_protocol	91
1.15.3	iso_8859_1	94
1.15.4	iso_8859_10	96
1.15.5	iso_8859_13	97
1.15.6	iso_8859_14	98
1.15.7	iso_8859_15	99
1.15.8	iso_8859_16	100
1.15.9	iso_8859_2	102
1.15.10	iso_8859_3	103
1.15.11	iso_8859_4	104
1.15.12	iso_8859_9	105
1.15.13	mapped_single_byte_character_set	106
1.15.14	single_byte_character_set(MaxCode)	108
1.15.15	us_ascii	109
1.15.16	utf_16_character_set(Endian)	110
1.15.17	utf_16be	112
1.15.18	utf_16le	113
1.15.19	utf_32_character_set(Endian)	114
1.15.20	utf_32be	115
1.15.21	utf_32le	116
1.15.22	utf_8	118
1.15.23	utf_8_character_set	119
1.15.24	windows_1250	120
1.15.25	windows_1251	121
1.15.26	windows_1252	122
1.15.27	windows_1253	124
1.15.28	windows_1254	125
1.15.29	windows_1257	126
1.16	classifier_protocols	127
1.16.1	classifier_protocol	127
1.16.2	dataset_protocol	130
1.17	code_metrics	133
1.17.1	cc_metric	133
1.17.2	code_metric	135
1.17.3	code_metrics	148
1.17.4	code_metrics_messages	149
1.17.5	code_metrics_utilities	150
1.17.6	cogc_metric	156
1.17.7	coupling_metric	157
1.17.8	dit_metric	159
1.17.9	doc_metric	161
1.17.10	halstead_metric	165
1.17.11	halstead_metric(Stroud)	166
1.17.12	lcom_metric	168
1.17.13	lines_metric	170
1.17.14	mi_metric	171
1.17.15	mi_metric(Stroud)	172
1.17.16	noc_metric	174
1.17.17	nor_metric	175
1.17.18	rfc_metric	177
1.17.19	size_metric	178
1.17.20	upn_metric	180
1.17.21	wmc_metric	181
1.18	combinations	183

1.18.1	combinations	183
1.18.2	combinations_protocol	184
1.19	command_line_options	192
1.19.1	command_line_option	192
1.19.2	command_line_options	197
1.20	core	200
1.20.1	core_messages	200
1.20.2	expanding	201
1.20.3	forwarding	203
1.20.4	logtalk	205
1.20.5	monitoring	220
1.20.6	user	222
1.21	coroutining	223
1.21.1	coroutining	223
1.22	csv	226
1.22.1	csv	227
1.22.2	csv(Header,Separator,IgnoreQuotes)	228
1.22.3	csv(Header,Separator,IgnoreQuotes,Comments)	229
1.22.4	csv_guess_questions	231
1.22.5	csv_protocol	232
1.23	cuid2	243
1.23.1	cuid2	243
1.23.2	cuid2(Representation,Size,Alphabet)	244
1.23.3	cuid2_protocol	246
1.24	datalog	247
1.24.1	datalog	247
1.24.2	datalog_protocol	257
1.25	dates	264
1.25.1	date	264
1.25.2	datep	266
1.25.3	time	277
1.25.4	timep	278
1.26	dates_tz	281
1.26.1	dates_tz	281
1.26.2	dates_tz_protocol	282
1.27	dead_code_scanner	286
1.27.1	dead_code_scanner	286
1.27.2	dead_code_scanner_messages	294
1.28	debug_messages	296
1.28.1	debug_messages	296
1.29	debugger	300
1.29.1	debugger	300
1.29.2	debugger_messages	310
1.29.3	debuggerp	311
1.29.4	dump_trace	323
1.30	dependents	325
1.30.1	observer	325
1.30.2	subject	326
1.31	deques	330
1.31.1	deque	330
1.31.2	deque_protocol	331
1.32	diagrams	337
1.32.1	caller_diagram	337
1.32.2	caller_diagram(Format)	338

1.32.3	cytoscapejs_graph_language	341
1.32.4	d2_graph_language	345
1.32.5	diagram(Format)	346
1.32.6	diagrams	370
1.32.7	diagrams(Format)	371
1.32.8	directory_dependency_diagram	381
1.32.9	directory_dependency_diagram(Format)	383
1.32.10	directory_diagram(Format)	384
1.32.11	directory_load_diagram	388
1.32.12	directory_load_diagram(Format)	389
1.32.13	dot_graph_language	391
1.32.14	entity_diagram	393
1.32.15	entity_diagram(Format)	394
1.32.16	file_dependency_diagram	398
1.32.17	file_dependency_diagram(Format)	399
1.32.18	file_diagram(Format)	401
1.32.19	file_load_diagram	405
1.32.20	file_load_diagram(Format)	406
1.32.21	graph_language_protocol	408
1.32.22	graph_language_registry	412
1.32.23	inheritance_diagram	413
1.32.24	inheritance_diagram(Format)	415
1.32.25	library_dependency_diagram	416
1.32.26	library_dependency_diagram(Format)	418
1.32.27	library_diagram(Format)	419
1.32.28	library_load_diagram	424
1.32.29	library_load_diagram(Format)	425
1.32.30	mermaid_graph_language	427
1.32.31	modules_diagram_support	429
1.32.32	plantuml_graph_language	431
1.32.33	uses_diagram	432
1.32.34	uses_diagram(Format)	434
1.32.35	xref_diagram	435
1.32.36	xref_diagram(Format)	437
1.33	dictionaries	440
1.33.1	avltree	440
1.33.2	bintree	442
1.33.3	dictionaryp	444
1.33.4	rbtree	457
1.33.5	splaytree	458
1.33.6	two3tree	460
1.34	dif	472
1.34.1	dif	472
1.35	doclet	474
1.35.1	doclet	474
1.36	edcg	477
1.36.1	edcg	477
1.37	events	480
1.37.1	after_event_registry	480
1.37.2	before_event_registry	482
1.37.3	event_registry	483
1.37.4	event_registryp	484
1.37.5	monitor	488
1.37.6	monitorp	490

1.38	expand_library_alias_paths	493
1.38.1	expand_library_alias_paths	493
1.39	expecteds	495
1.39.1	either	495
1.39.2	expected	498
1.39.3	expected(Expected)	504
1.40	fcube	517
1.40.1	fcube	517
1.41	flags	521
1.41.1	flags	521
1.41.2	flags_validator	528
1.42	format	530
1.42.1	format	530
1.43	genint	532
1.43.1	genint	532
1.43.2	genint_core	534
1.44	gensym	536
1.44.1	gensym	536
1.44.2	gensym_core	537
1.45	geospatial	540
1.45.1	geospatial	540
1.45.2	geospatial_protocol	542
1.46	git	566
1.46.1	git	566
1.46.2	git_protocol	567
1.47	grammars	571
1.47.1	blank_grammars(Format)	571
1.47.2	ip_grammars(Format)	578
1.47.3	number_grammars(Format)	579
1.47.4	sequence_grammars	585
1.48	graphs	589
1.48.1	directed_graph_common	589
1.48.2	directed_graph_protocol	590
1.48.3	graph_common	596
1.48.4	graph_protocol	599
1.48.5	graph_types	612
1.48.6	undirected_graph_common	613
1.48.7	unweighted_directed_graph	618
1.48.8	unweighted_directed_graph(Dictionary)	619
1.48.9	unweighted_graph_common(Dictionary)	622
1.48.10	unweighted_graph_protocol	624
1.48.11	unweighted_undirected_graph	627
1.48.12	unweighted_undirected_graph(Dictionary)	628
1.48.13	weighted_directed_graph	630
1.48.14	weighted_directed_graph(Dictionary)	632
1.48.15	weighted_graph_common(Dictionary)	634
1.48.16	weighted_graph_protocol	637
1.48.17	weighted_undirected_graph	641
1.48.18	weighted_undirected_graph(Dictionary)	642
1.49	hashes	645
1.49.1	crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength)	645
1.49.2	crc32_reflected(Polynomial)	647
1.49.3	crc32b	648
1.49.4	crc32bzip2	650

1.49.5	crc32c	651
1.49.6	crc32mpeg2	652
1.49.7	crc32posix	654
1.49.8	crc32q	655
1.49.9	djb2_32	656
1.49.10	djb2_64	658
1.49.11	fips202_hash(A,B,C)	659
1.49.12	fnv1a_32	660
1.49.13	fnv1a_64	662
1.49.14	hash_common_32	663
1.49.15	hash_common_64	669
1.49.16	hash_protocol	675
1.49.17	md5	676
1.49.18	murmurhash3_x64_128	678
1.49.19	murmurhash3_x86_128	679
1.49.20	murmurhash3_x86_32	681
1.49.21	sdbm_32	682
1.49.22	sdbm_64	683
1.49.23	sha1	685
1.49.24	sha256	686
1.49.25	sha3_224	687
1.49.26	sha3_256	689
1.49.27	sha3_384	690
1.49.28	sha3_512	691
1.49.29	shake128(OutputBytes)	693
1.49.30	shake256(OutputBytes)	694
1.49.31	siphash_2_4	695
1.49.32	siphash_2_4(Key)	697
1.50	heaps	698
1.50.1	binary_heap(Order)	698
1.50.2	binary_heap_max	700
1.50.3	binary_heap_min	701
1.50.4	heap(Order)	702
1.50.5	heap_protocol	704
1.50.6	heapp	709
1.50.7	maxheap	710
1.50.8	minheap	711
1.50.9	pairing_heap(Order)	713
1.50.10	pairing_heap_max	714
1.50.11	pairing_heap_min	716
1.51	help	717
1.51.1	help	717
1.52	hierarchies	725
1.52.1	class_hierarchy	726
1.52.2	class_hierarchy_p	727
1.52.3	hierarchyp	734
1.52.4	proto_hierarchy	737
1.52.5	proto_hierarchy_p	739
1.53	hook_flows	741
1.53.1	hook_pipeline(Pipeline)	742
1.53.2	hook_set(Set)	743
1.54	hook_objects	744
1.54.1	backend_adapter_hook	745
1.54.2	default_workflow_hook	746

1.54.3	grammar_rules_hook	747
1.54.4	identity_hook	749
1.54.5	object_wrapper_hook	750
1.54.6	object_wrapper_hook(Protocol)	751
1.54.7	object_wrapper_hook(Name,Relations)	753
1.54.8	print_goal_hook	754
1.54.9	prolog_module_hook(Module)	756
1.54.10	suppress_goal_hook	757
1.54.11	write_to_file_hook(File)	758
1.54.12	write_to_file_hook(File,Options)	760
1.54.13	write_to_stream_hook(Stream)	761
1.54.14	write_to_stream_hook(Stream,Options)	763
1.55	html	764
1.55.1	html	764
1.55.2	html5	767
1.55.3	xhtml11	768
1.56	ids	769
1.56.1	ids	769
1.56.2	ids(Representation,Bytes)	771
1.57	intervals	772
1.57.1	interval	773
1.57.2	intervalp	774
1.58	iso8601	781
1.58.1	iso8601	781
1.59	isolation_forest	796
1.59.1	isolation_forest	796
1.60	issue_creator	800
1.60.1	issue_creator	800
1.61	java	801
1.61.1	java	801
1.61.2	java(Reference)	803
1.61.3	java(Reference,ReturnValue)	804
1.61.4	java_access_protocol	806
1.61.5	java_hook	809
1.61.6	java_utils_protocol	810
1.62	json	820
1.62.1	json	820
1.62.2	json(StringRepresentation)	821
1.62.3	json(ObjectRepresentation,PairRepresentation,StringRepresentation)	822
1.62.4	json_protocol	824
1.63	json_ld	825
1.63.1	json_ld	826
1.63.2	json_ld(StringRepresentation)	827
1.63.3	json_ld(ObjectRepresentation,PairRepresentation,StringRepresentation)	828
1.63.4	json_ld_protocol	830
1.64	json_lines	833
1.64.1	json_lines	834
1.64.2	json_lines(StringRepresentation)	835
1.64.3	json_lines(ObjectRepresentation,PairRepresentation,StringRepresentation)	836
1.64.4	json_lines_protocol	838
1.65	json_rpc	840
1.65.1	json_rpc	840
1.66	json_schema	853
1.66.1	json_schema	854

1.66.2	json_schema(StringRepresentation)	855
1.66.3	json_schema(ObjectRepresentation,PairRepresentation,StringRepresentation)	856
1.66.4	json_schema_protocol	858
1.67	knn	860
1.67.1	knn	860
1.68	ksuid	863
1.68.1	ksuid	863
1.68.2	ksuid(Representation,Alphabet)	864
1.68.3	ksuid_protocol	866
1.69	lgtdoc	868
1.69.1	lgtdoc	868
1.69.2	lgtdoc_messages	872
1.69.3	lgtdocp	873
1.70	lgtunit	885
1.70.1	automation_report	885
1.70.2	cobertura_report	886
1.70.3	coverage_report	890
1.70.4	ctrf_output	892
1.70.5	ctrf_report	893
1.70.6	lcov_report	895
1.70.7	lgtunit	897
1.70.8	lgtunit_messages	947
1.70.9	minimal_output	949
1.70.10	subunit_v1_output	950
1.70.11	subunit_v1_report	951
1.70.12	subunit_v2_output	953
1.70.13	subunit_v2_report	954
1.70.14	tap_output	955
1.70.15	tap_report	958
1.70.16	xunit_net_v2_output	960
1.70.17	xunit_net_v2_report	961
1.70.18	xunit_output	963
1.70.19	xunit_report	965
1.71	library	966
1.71.1	cloning	966
1.71.2	counters	968
1.71.3	streamvars	971
1.72	linda	974
1.72.1	linda	974
1.72.2	linda_client	976
1.72.3	linda_server	989
1.73	linter_reporter	997
1.73.1	linter_reporter	997
1.74	listing	1003
1.74.1	listing	1003
1.75	logging	1005
1.75.1	logger	1006
1.75.2	logging	1008
1.75.3	loggingp	1010
1.76	loops	1014
1.76.1	loop	1014
1.76.2	loopp	1015
1.77	mcp_server	1021
1.77.1	mcp_prompt_protocol	1021

1.77.2	mcp_resource_protocol	1023
1.77.3	mcp_server	1025
1.77.4	mcp_tool_protocol	1032
1.78	memcached	1036
1.78.1	memcached	1036
1.79	meta	1049
1.79.1	meta	1049
1.79.2	metap	1051
1.80	meta_compiler	1061
1.80.1	meta_compiler	1062
1.81	metagol	1063
1.81.1	metagol	1064
1.81.2	metagol_example_protocol	1071
1.82	mime_types	1073
1.82.1	mime_types	1073
1.83	mutation_testing	1081
1.83.1	arithmetic_operator_replacement(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)	1081
1.83.2	body_goal_negation(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)	1083
1.83.3	clause_mutator_protocol	1084
1.83.4	clauses_reordering(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)	1086
1.83.5	directive_mutator_protocol	1087
1.83.6	fail_insertion(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)	1089
1.83.7	head_arguments_mutation(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)	1090
1.83.8	head_arguments_reordering(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)	1092
1.83.9	mutation_testing	1093
1.83.10	mutation_testing_messages	1105
1.83.11	mutator_common	1106
1.83.12	mutator_protocol	1114
1.83.13	predicate_directive_suppression(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)	1117
1.83.14	relational_operator_replacement(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)	1118
1.83.15	scope_directive_replacement(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)	1120
1.83.16	subprocess_coverage_hook	1121
1.83.17	subprocess_mutation_hook	1123
1.83.18	truth_literal_flip(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)	1126
1.83.19	uses_directive_resource_deletion(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation)	1127
1.84	mutations	1128
1.84.1	default_atom_mutations	1129
1.84.2	default_compound_mutations	1130
1.84.3	default_float_mutations	1131
1.84.4	default_integer_mutations	1133
1.84.5	default_list_mutations	1134
1.84.6	mutations	1135
1.84.7	mutations_store	1137
1.85	naive_bayes	1140
1.85.1	naive_bayes	1140
1.86	nanoid	1142
1.86.1	nanoid	1142
1.86.2	nanoid(Representation,Size,Alphabet)	1143

1.86.3	nanoid_protocol	1145
1.87	nearest_centroid	1146
1.87.1	nearest_centroid	1146
1.88	nested_dictionaries	1149
1.88.1	navltree	1149
1.88.2	nbintree	1151
1.88.3	nested_dictionary_protocol	1152
1.88.4	nrbtree	1157
1.89	optionals	1158
1.89.1	maybe	1158
1.89.2	optional	1161
1.89.3	optional(Optional)	1166
1.90	options	1176
1.90.1	options	1176
1.90.2	options_protocol	1177
1.91	os	1183
1.91.1	os	1183
1.91.2	os_types	1185
1.91.3	osp	1187
1.92	packs	1205
1.92.1	pack_protocol	1206
1.92.2	packs	1210
1.92.3	packs_common	1246
1.92.4	packs_messages	1258
1.92.5	packs_specs_hook	1259
1.92.6	registries	1261
1.92.7	registry_loader_hook	1273
1.92.8	registry_protocol	1274
1.93	pddl_parser	1278
1.93.1	pddl	1278
1.93.2	read_file	1281
1.94	permutations	1282
1.94.1	permutations	1282
1.94.2	permutations_protocol	1284
1.95	ports_profiler	1293
1.95.1	ports_profiler	1293
1.96	process	1299
1.96.1	process	1299
1.97	protobuf	1302
1.97.1	protobuf	1302
1.98	queues	1305
1.98.1	queue	1305
1.98.2	queup	1306
1.99	random	1313
1.99.1	backend_random	1313
1.99.2	fast_random	1315
1.99.3	fast_random(Algorithm)	1316
1.99.4	pseudo_random_protocol	1319
1.99.5	random	1321
1.99.6	random(Algorithm)	1323
1.99.7	random_protocol	1326
1.99.8	sampling_protocol	1335
1.100	random_forest	1348
1.100.1	random_forest	1348

1.101	reader	1350
1.101.1	reader	1351
1.102	recorded_database	1360
1.102.1	recorded_database	1360
1.102.2	recorded_database_core	1361
1.103	redis	1367
1.103.1	redis	1367
1.104	sarif	1388
1.104.1	sarif	1388
1.105	sbom	1391
1.105.1	sbom	1392
1.106	sets	1395
1.106.1	set	1395
1.106.2	set(Type)	1397
1.106.3	setp	1399
1.106.4	treap_set	1408
1.107	simulated_annealing	1410
1.107.1	simulated_annealing(Problem)	1410
1.107.2	simulated_annealing(Problem,RandomAlgorithm)	1411
1.107.3	simulated_annealing_protocol	1416
1.108	snowflakeid	1421
1.108.1	snowflakeid	1421
1.108.2	snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds, TimestampBits,NodeBits,SequenceBits,Node)	1422
1.108.3	snowflakeid_instagram	1424
1.108.4	snowflakeid_instagram(Representation)	1425
1.108.5	snowflakeid_protocol	1427
1.108.6	snowflakeid_sonyflake	1428
1.108.7	snowflakeid_sonyflake(Representation)	1429
1.108.8	snowflakeid_twitter	1431
1.108.9	snowflakeid_twitter(Representation)	1432
1.109	sockets	1433
1.109.1	socket	1433
1.110	statistics	1440
1.110.1	population	1440
1.110.2	sample	1442
1.110.3	statistics	1443
1.110.4	statisticsp	1449
1.111	stemming	1465
1.111.1	lovins_stemmer(Representation)	1466
1.111.2	porter_stemmer(Representation)	1467
1.111.3	stemmer_protocol	1469
1.112	stomp	1471
1.112.1	stomp	1471
1.113	string_distance	1480
1.113.1	string_distance(Representation)	1480
1.114	strings	1490
1.114.1	string(Representation)	1490
1.115	subsequences	1498
1.115.1	subsequences	1498
1.115.2	subsequences_protocol	1500
1.116	term_io	1522
1.116.1	term_io	1523
1.116.2	term_io_protocol	1524

1.117	time_scales	1535
1.117.1	time_scales	1535
1.117.2	time_scales_data	1536
1.117.3	time_scales_protocol	1545
1.118	timeout	1557
1.118.1	timeout	1557
1.119	toml	1559
1.119.1	toml	1560
1.119.2	toml(StringRepresentation)	1561
1.119.3	toml(ObjectRepresentation,PairRepresentation,StringRepresentation)	1562
1.119.4	toml_protocol	1564
1.120	tool_diagnostics	1566
1.120.1	tool_diagnostics_common	1566
1.120.2	tool_diagnostics_protocol	1568
1.121	toon	1574
1.121.1	toon	1574
1.121.2	toon(StringRepresentation)	1575
1.121.3	toon(ObjectRepresentation,PairRepresentation,StringRepresentation)	1576
1.121.4	toon_protocol	1578
1.122	toychr	1580
1.122.1	toychrdb	1580
1.123	tsv	1585
1.123.1	tsv	1586
1.123.2	tsv(Header)	1587
1.123.3	tsv(Header,Comments)	1588
1.123.4	tsv_protocol	1590
1.124	tutor	1599
1.124.1	tutor	1599
1.124.2	tutor_explanations	1600
1.125	types	1602
1.125.1	atom	1602
1.125.2	atomic	1604
1.125.3	callable	1605
1.125.4	character	1606
1.125.5	characterp	1608
1.125.6	comparingp	1617
1.125.7	compound	1620
1.125.8	difflist	1621
1.125.9	float	1624
1.125.10	integer	1626
1.125.11	list	1630
1.125.12	list(Type)	1632
1.125.13	listp	1633
1.125.14	natural	1656
1.125.15	number	1658
1.125.16	numberlist	1662
1.125.17	numberlistp	1664
1.125.18	pairs	1675
1.125.19	term	1680
1.125.20	termp	1681
1.125.21	type	1688
1.125.22	varlist	1694
1.125.23	varlistp	1695
1.126	tzif	1705

1.126.1	tzif	1705
1.126.2	tzif_protocol	1707
1.126.3	tzif_zone_ids	1733
1.127	ulid	1735
1.127.1	ulid	1735
1.127.2	ulid(Representation)	1736
1.127.3	ulid_protocol	1738
1.127.4	ulid_types	1741
1.128	union_find	1742
1.128.1	union_find	1742
1.128.2	union_find_protocol	1744
1.129	url	1748
1.129.1	url(Representation)	1748
1.130	uuid	1752
1.130.1	uuid	1752
1.130.2	uuid(Representation)	1753
1.130.3	uuid_protocol	1755
1.131	validations	1759
1.131.1	validated	1759
1.131.2	validation	1764
1.131.3	validation(Validation)	1770
1.132	verdi_neruda	1783
1.132.1	a_star_interpreter(W)	1784
1.132.2	benchmark_generators	1785
1.132.3	best_first	1786
1.132.4	bfs_interpreter	1788
1.132.5	bup_interpreter	1789
1.132.6	counter	1790
1.132.7	databasep	1794
1.132.8	debug_expansion(Mode)	1796
1.132.9	demodb	1798
1.132.10	dfs_interpreter	1799
1.132.11	flatting	1800
1.132.12	heuristic_expansion(Mode)	1802
1.132.13	iddfs_interpreter(Increment)	1803
1.132.14	interpreterp	1804
1.132.15	magic	1806
1.132.16	magic_expansion(Mode)	1808
1.132.17	rule_expansion(Mode)	1809
1.132.18	shell	1810
1.132.19	shell(Interpreters)	1812
1.132.20	shell_expansion(Mode)	1813
1.133	wrapper	1815
1.133.1	wrapper	1815
1.134	xml_parser	1827
1.134.1	xml	1827
1.135	yaml	1835
1.135.1	yaml	1835
1.135.2	yaml_protocol	1836
1.136	zippers	1840
1.136.1	zipperp	1840
1.136.2	zlist	1850

2.1	contributions/flags/	1855
2.2	contributions/iso8601/	1855
2.3	contributions/pddl_parser/	1855
2.4	contributions/verdi_neruda/	1855
2.5	contributions/xml_parser/	1855
2.6	core/	1855
2.7	library/	1855
2.8	library/ada_boost/	1855
2.9	library/amqp/	1855
2.10	library/application/	1855
2.11	library/arbitrary/	1855
2.12	library/assignvars/	1855
2.13	library/avro/	1855
2.14	library/base32/	1855
2.15	library/base58/	1855
2.16	library/base64/	1855
2.17	library/base85/	1855
2.18	library/c45/	1855
2.19	library/cbor/	1855
2.20	library/ccsds/	1855
2.21	library/character_sets/	1855
2.22	library/classifier_protocols/	1855
2.23	library/combinations/	1855
2.24	library/command_line_options/	1855
2.25	library/coroutining/	1855
2.26	library/csv/	1855
2.27	library/cuid2/	1855
2.28	library/datalog/	1855
2.29	library/dates/	1855
2.30	library/dates_tz/	1855
2.31	library/dependents/	1855
2.32	library/deques/	1855
2.33	library/dictionaries/	1855
2.34	library/dif/	1855
2.35	library/edcg/	1855
2.36	library/events/	1855
2.37	library/expand_library_alias_paths/	1855
2.38	library/expecteds/	1855
2.39	library/format/	1855
2.40	library/genint/	1855
2.41	library/gensym/	1855
2.42	library/geospatial/	1855
2.43	library/git/	1855
2.44	library/grammars/	1855
2.45	library/graphs/	1855
2.46	library/hashes/	1855
2.47	library/heaps/	1855
2.48	library/hierarchies/	1855
2.49	library/hook_flows/	1855
2.50	library/hook_objects/	1855
2.51	library/html/	1855
2.52	library/ids/	1855
2.53	library/intervals/	1855
2.54	library/isolation_forest/	1855

2.55	library/java/	1855
2.56	library/json/	1855
2.57	library/json_ld/	1855
2.58	library/json_lines/	1855
2.59	library/json_rpc/	1855
2.60	library/json_schema/	1855
2.61	library/knn/	1855
2.62	library/ksuid/	1855
2.63	library/linda/	1855
2.64	library/listing/	1855
2.65	library/logging/	1855
2.66	library/loops/	1855
2.67	library/mcp_server/	1855
2.68	library/memcached/	1855
2.69	library/meta/	1855
2.70	library/meta_compiler/	1855
2.71	library/mime_types/	1855
2.72	library/mutations/	1855
2.73	library/naive_bayes/	1855
2.74	library/nanoid/	1855
2.75	library/nearest_centroid/	1855
2.76	library/nested_dictionaries/	1855
2.77	library/optionals/	1855
2.78	library/options/	1855
2.79	library/os/	1855
2.80	library/permutations/	1855
2.81	library/process/	1855
2.82	library/protobuf/	1855
2.83	library/queues/	1855
2.84	library/random/	1855
2.85	library/random_forest/	1855
2.86	library/reader/	1855
2.87	library/recorded_database/	1855
2.88	library/redis/	1855
2.89	library/sets/	1855
2.90	library/simulated_annealing/	1855
2.91	library/snowflakeid/	1855
2.92	library/sockets/	1855
2.93	library/statistics/	1855
2.94	library/stemming/	1855
2.95	library/stomp/	1855
2.96	library/string_distance/	1855
2.97	library/strings/	1855
2.98	library/subsequences/	1855
2.99	library/term_io/	1855
2.100	library/time_scales/	1855
2.101	library/timeout/	1855
2.102	library/toml/	1855
2.103	library/toon/	1855
2.104	library/tsv/	1855
2.105	library/types/	1855
2.106	library/tzif/	1855
2.107	library/ulid/	1855
2.108	library/union_find/	1855

2.109	library/url/	1855
2.110	library/uuid/	1855
2.111	library/validations/	1855
2.112	library/yaml/	1855
2.113	library/zippers/	1855
2.114	ports/fcube/	1855
2.115	ports/metagol/	1855
2.116	ports/toychr/	1855
2.117	tools/assertions/	1855
2.118	tools/code_metrics/	1855
2.119	tools/dead_code_scanner/	1855
2.120	tools/debug_messages/	1855
2.121	tools/debugger/	1855
2.122	tools/diagrams/	1855
2.123	tools/doclet/	1855
2.124	tools/help/	1855
2.125	tools/issue_creator/	1855
2.126	tools/lgtdoc/	1855
2.127	tools/lgtunit/	1855
2.128	tools/linter_reporter/	1855
2.129	tools/mutation_testing/	1855
2.130	tools/mutation_testing/mutators/	1855
2.131	tools/packs/	1855
2.132	tools/ports_profiler/	1855
2.133	tools/sarif/	1855
2.134	tools/sbom/	1855
2.135	tools/tool_diagnostics/	1855
2.136	tools/tutor/	1855
2.137	tools/wrapper/	1855
3	Entities	1857
3.1	Categories	1857
3.2	Objects	1857
3.3	Protocols	1857
4	Predicates	1859
4.1	(/)/2	1859
4.2	(//)/2	1859
4.3	(<)/2	1859
4.4	(<=)/2	1859
4.5	(=:=)/2	1859
4.6	(=<)/2	1860
4.7	(=>)/2	1860
4.8	(=\=)/2	1860
4.9	=~/2	1860
4.10	(>)/2	1860
4.11	(>=)/2	1860
4.12	abbreviation/2	1860
4.13	abbreviation/3	1860
4.14	abbreviation/4	1861
4.15	abort_transaction/3	1861
4.16	absolute_file_name/2	1861
4.17	ack/3	1861
4.18	acquire/1	1861

4.19	activate_debug_handler/1	1861
4.20	activate_monitor/0	1861
4.21	active_debug_handler/1	1861
4.22	add/1	1862
4.23	add/2	1862
4.24	add/3	1862
4.25	add/5	1862
4.26	add32/3	1862
4.27	add32/4	1862
4.28	add32/5	1862
4.29	add64/3	1862
4.30	addDependent/1	1863
4.31	add_duration/3	1863
4.32	add_edge/4	1863
4.33	add_edge/5	1863
4.34	add_edges/3	1863
4.35	add_rule/3	1863
4.36	add_type/2	1863
4.37	add_type/3	1863
4.38	add_vertex/3	1864
4.39	add_vertices/3	1864
4.40	after/2	1864
4.41	after/3	1864
4.42	alias/1	1864
4.43	all/0	1864
4.44	all/1	1864
4.45	all_files/0	1865
4.46	all_files/1	1865
4.47	all_libraries/0	1865
4.48	all_libraries/1	1865
4.49	all_pairs_min_paths/2	1865
4.50	all_pairs_min_predecessors/2	1865
4.51	all_score/1	1865
4.52	along_track_distance/4	1866
4.53	alternating_subsequence/2	1866
4.54	alternating_subsequences/2	1866
4.55	ancestor/1	1866
4.56	ancestors/1	1866
4.57	and64/3	1866
4.58	apid/2	1866
4.59	apis/0	1866
4.60	apis/1	1867
4.61	append/2	1867
4.62	append/3	1867
4.63	append/4	1867
4.64	apply/2	1867
4.65	apply/4	1867
4.66	approximately_equal/2	1867
4.67	approximately_equal/3	1868
4.68	arbitrary/1	1868
4.69	arbitrary/2	1868
4.70	archive/1	1868
4.71	arithmetic_mean/2	1868
4.72	array_list/2	1868

4.73	array_to_list/2	1868
4.74	array_to_terms/2	1868
4.75	array_to_terms/3	1869
4.76	articulation_points/2	1869
4.77	as_curly_bracketed/2	1869
4.78	as_deque/2	1869
4.79	as_dictionary/2	1869
4.80	as_difflist/2	1869
4.81	as_heap/2	1869
4.82	as_list/2	1869
4.83	as_nested_dictionary/2	1870
4.84	as_set/2	1870
4.85	ask_question/5	1870
4.86	assert_fact/1	1870
4.87	assertion/1	1870
4.88	assertion/2	1870
4.89	assignable/1	1870
4.90	assignable/2	1870
4.91	atom_string/2	1871
4.92	atomics_to_string/2	1871
4.93	atomics_to_string/3	1871
4.94	attribute_values/2	1871
4.95	available/0	1871
4.96	available/1	1871
4.97	available/2	1871
4.98	average/2	1871
4.99	average_deviation/3	1872
4.100	basic_ack/3	1872
4.101	basic_cancel/3	1872
4.102	basic_consume/3	1872
4.103	basic_get/3	1872
4.104	basic_nack/3	1872
4.105	basic_publish/4	1872
4.106	basic_qos/2	1872
4.107	basic_recover/2	1873
4.108	basic_reject/3	1873
4.109	bbox_contains/2	1873
4.110	bbox_expand/3	1873
4.111	bbox_from_coordinates/2	1873
4.112	bbox_intersects/2	1873
4.113	bbox_union/3	1873
4.114	before/2	1873
4.115	before/3	1874
4.116	begin/0	1874
4.117	begin_transaction/3	1874
4.118	bench_goal/1	1874
4.119	benchmark/2	1874
4.120	benchmark/3	1874
4.121	benchmark/4	1874
4.122	benchmark_reified/3	1874
4.123	bernoulli/2	1875
4.124	beta/3	1875
4.125	between/3	1875
4.126	between/4	1875

4.127	big_endian_word32/2	1875
4.128	binomial/3	1875
4.129	bit//1	1875
4.130	bits//1	1875
4.131	blank//0	1876
4.132	blanks//0	1876
4.133	body_pred/1	1876
4.134	bounding_box/3	1876
4.135	branch/2	1876
4.136	breadth_first_order/3	1876
4.137	bridges/2	1876
4.138	built_date/1	1876
4.139	built_in_directive/4	1877
4.140	built_in_flag/2	1877
4.141	built_in_method/4	1877
4.142	built_in_non_terminal/4	1877
4.143	built_in_predicate/4	1877
4.144	bytes_hex/2	1877
4.145	bytes_to_codes/2	1877
4.146	cache/1	1877
4.147	cache_source/1	1878
4.148	cached_tzif/1	1878
4.149	calendar_month/3	1878
4.150	call_with_timeout/2	1878
4.151	call_with_timeout/3	1878
4.152	capabilities/1	1878
4.153	cartesian_product/3	1878
4.154	cas/6	1878
4.155	cat/2	1879
4.156	central_moment/3	1879
4.157	change_directory/1	1879
4.158	changed/0	1879
4.159	changed/1	1879
4.160	channel_close/1	1879
4.161	channel_close/3	1879
4.162	channel_open/3	1879
4.163	chebyshev_distance/3	1880
4.164	chebyshev_norm/2	1880
4.165	check/0	1880
4.166	check/1	1880
4.167	check/2	1880
4.168	check/3	1880
4.169	check_conversion/3	1880
4.170	check_convert/4	1880
4.171	check_instant/1	1881
4.172	check_offset/3	1881
4.173	check_option/1	1881
4.174	check_options/1	1881
4.175	chi_squared/2	1881
4.176	chr_is/2	1881
4.177	chr_no_spy/1	1881
4.178	chr_nospy/0	1881
4.179	chr_notrace/0	1882
4.180	chr_option/2	1882

4.181	chr_spy/1	1882
4.182	chr_trace/0	1882
4.183	circular_uniform_cartesian/3	1882
4.184	circular_uniform_polar/3	1882
4.185	class/1	1882
4.186	class_values/1	1882
4.187	classes/1	1883
4.188	classifier_to_clauses/4	1883
4.189	classifier_to_file/4	1883
4.190	clause/5	1883
4.191	clause_location/6	1883
4.192	clean/0	1883
4.193	clean/1	1883
4.194	clean/2	1883
4.195	clear/0	1884
4.196	clear_cache/0	1884
4.197	clear_dut1_override/0	1884
4.198	clear_leap_seconds_override/0	1884
4.199	client_open/4	1884
4.200	client_open/5	1884
4.201	clockwise_polygon/2	1884
4.202	clone/1	1884
4.203	clone/3	1885
4.204	clone/4	1885
4.205	close/1	1885
4.206	close/2	1885
4.207	close/3	1885
4.208	close_client/1	1885
4.209	close_polygon/2	1885
4.210	codes_to_bytes/2	1885
4.211	coefficient_of_variation/2	1886
4.212	combination/3	1886
4.213	combination/4	1886
4.214	combination_index/4	1886
4.215	combination_with_replacement/3	1886
4.216	combination_with_replacement/4	1886
4.217	combinations/3	1886
4.218	combinations/4	1886
4.219	combinations_with_replacement/3	1887
4.220	combinations_with_replacement/4	1887
4.221	command_line_arguments/1	1887
4.222	commit/0	1887
4.223	commit_author/2	1887
4.224	commit_date/2	1887
4.225	commit_hash/2	1887
4.226	commit_hash_abbreviated/2	1887
4.227	commit_log/3	1888
4.228	commit_message/2	1888
4.229	commit_transaction/3	1888
4.230	common_subsequence/3	1888
4.231	common_subsequences/3	1888
4.232	compact/3	1888
4.233	compare_date_time/3	1888
4.234	compile_aux_clauses/1	1888

4.235	compile_predicate_heads/4	1889
4.236	compile_predicate_indicators/3	1889
4.237	complement/2	1889
4.238	completion/2	1889
4.239	completions/2	1889
4.240	compose/3	1889
4.241	confirm_select/1	1889
4.242	connect/1	1889
4.243	connect/2	1890
4.244	connect/3	1890
4.245	connect/4	1890
4.246	connected_components/2	1890
4.247	connection_alive/1	1890
4.248	console/1	1890
4.249	contains/2	1890
4.250	control//0	1891
4.251	control_construct/4	1891
4.252	controls//0	1891
4.253	convert/4	1891
4.254	convert_zones/4	1891
4.255	convert_zones_with_resolution/5	1891
4.256	cooling_schedule/3	1891
4.257	coordinates_bounding_box/2	1891
4.258	copy_file/2	1892
4.259	correlation/3	1892
4.260	cosine_similarity/3	1892
4.261	count_combinations/3	1892
4.262	count_combinations_with_replacement/3	1892
4.263	count_distinct_subsequences/3	1892
4.264	count_permutations/2	1892
4.265	count_subsequences/2	1892
4.266	counter/2	1893
4.267	counterclockwise_polygon/2	1893
4.268	covariance/3	1893
4.269	cover/1	1893
4.270	coverage_clause_mutator/0	1893
4.271	cpu_time/1	1893
4.272	create/3	1893
4.273	creators/1	1893
4.274	cross_track_distance/4	1894
4.275	current/2	1894
4.276	current_host/1	1894
4.277	cycle/2	1894
4.278	damerau_levenshtein/3	1894
4.279	data/0	1894
4.280	data/1	1894
4.281	data/2	1894
4.282	data_length/2	1895
4.283	date/4	1895
4.284	date/5	1895
4.285	date/6	1895
4.286	date/7	1895
4.287	date_string/3	1895
4.288	date_time/7	1895

4.289	date_time_string/3	1895
4.290	date_time_to_unix/2	1896
4.291	day_of_year/2	1896
4.292	day_of_year_date/3	1896
4.293	daylight_saving_time/2	1896
4.294	daylight_saving_time/3	1896
4.295	daylight_saving_time/4	1896
4.296	days_in_month/3	1896
4.297	deactivate_debug_handler/0	1896
4.298	debug/0	1897
4.299	debug_handler/1	1897
4.300	debug_handler/3	1897
4.301	debugging/0	1897
4.302	debugging/1	1897
4.303	decide/1	1897
4.304	decide/2	1897
4.305	decode/2	1897
4.306	decode_exception/2	1898
4.307	decode_exception/3	1898
4.308	decode_frame/2	1898
4.309	decompile_predicate_heads/4	1898
4.310	decompile_predicate_indicators/4	1898
4.311	decompose_file_name/3	1898
4.312	decompose_file_name/4	1898
4.313	decr/3	1898
4.314	decr/4	1899
4.315	decrby/4	1899
4.316	decrement_counter/1	1899
4.317	default/1	1899
4.318	default_option/1	1899
4.319	default_options/1	1899
4.320	define_log_file/2	1899
4.321	defined/4	1899
4.322	defined_flag/6	1900
4.323	degree/3	1900
4.324	del/3	1900
4.325	del_monitors/0	1900
4.326	del_monitors/4	1900
4.327	del_spy_points/4	1900
4.328	delete/0	1900
4.329	delete/1	1900
4.330	delete/2	1901
4.331	delete/3	1901
4.332	delete/4	1901
4.333	delete_all_after/2	1901
4.334	delete_all_after_and_unzip/2	1901
4.335	delete_all_before/2	1901
4.336	delete_all_before_and_unzip/2	1901
4.337	delete_and_next/2	1902
4.338	delete_and_previous/2	1902
4.339	delete_and_unzip/2	1902
4.340	delete_directory/1	1902
4.341	delete_directory_and_contents/1	1902
4.342	delete_directory_contents/1	1902

4.343 delete_edge/4	1902
4.344 delete_edge/5	1902
4.345 delete_edges/3	1903
4.346 delete_file/1	1903
4.347 delete_in/4	1903
4.348 delete_matches/3	1903
4.349 delete_max/4	1903
4.350 delete_min/4	1903
4.351 delete_vertex/3	1903
4.352 delete_vertices/3	1903
4.353 dependents/1	1904
4.354 dependents/2	1904
4.355 dependents/3	1904
4.356 depth/2	1904
4.357 depth_first_order/3	1904
4.358 derangement/2	1904
4.359 derangements/2	1904
4.360 descendant/1	1904
4.361 descendant_class/1	1905
4.362 descendant_classes/1	1905
4.363 descendant_instance/1	1905
4.364 descendant_instances/1	1905
4.365 descendants/1	1905
4.366 describe/1	1905
4.367 describe/2	1905
4.368 description/1	1905
4.369 destination_point/4	1906
4.370 destroy/0	1906
4.371 deterministic/1	1906
4.372 deterministic/2	1906
4.373 diagnostic/2	1906
4.374 diagnostic/3	1906
4.375 diagnostic_rule/5	1906
4.376 diagnostic_rules/1	1906
4.377 diagnostic_target/1	1907
4.378 diagnostics/2	1907
4.379 diagnostics/3	1907
4.380 diagnostics_preflight/2	1907
4.381 diagnostics_preflight/3	1907
4.382 diagnostics_summary/2	1907
4.383 diagnostics_summary/3	1907
4.384 diagnostics_tool/5	1907
4.385 diagram_description/1	1908
4.386 diagram_name_suffix/1	1908
4.387 dif/1	1908
4.388 dif/2	1908
4.389 digit//1	1908
4.390 digits//1	1908
4.391 directories/1	1908
4.392 directories/2	1909
4.393 directories/3	1909
4.394 directory/1	1909
4.395 directory/2	1909
4.396 directory/3	1910

4.397	directory_exists/1	1910
4.398	directory_files/2	1910
4.399	directory_files/3	1910
4.400	directory_mutants/2	1910
4.401	directory_mutants/3	1910
4.402	directory_score/2	1910
4.403	dirichlet/2	1910
4.404	disable/0	1911
4.405	disable/1	1911
4.406	disable/2	1911
4.407	disable_logging/1	1911
4.408	disconnect/1	1911
4.409	disconnect/2	1911
4.410	disjoint/2	1911
4.411	disjoint_sets/2	1911
4.412	distance/4	1912
4.413	distance/5	1912
4.414	distinct_combination/3	1912
4.415	distinct_combination/4	1912
4.416	distinct_combinations/3	1912
4.417	distinct_combinations/4	1912
4.418	distinct_permutation/2	1912
4.419	distinct_permutation/3	1912
4.420	distinct_permutations/2	1913
4.421	distinct_permutations/3	1913
4.422	distribution/1	1913
4.423	doc_goal/1	1913
4.424	document/1	1913
4.425	document/2	1913
4.426	dot//1	1913
4.427	double_metaphone/3	1913
4.428	double_metaphone_match/2	1914
4.429	dowhile/2	1914
4.430	drop/3	1914
4.431	duration_between/3	1914
4.432	duration_string/2	1914
4.433	during/2	1914
4.434	dut1_entries/1	1914
4.435	dut1_offset_at_utc_unix/3	1914
4.436	dut1_source/1	1915
4.437	easter_day/3	1915
4.438	edge/3	1915
4.439	edge/4	1915
4.440	edge/6	1915
4.441	edge_case/2	1915
4.442	edges/2	1915
4.443	edit_similarity/3	1915
4.444	edit_similarity/4	1916
4.445	either/3	1916
4.446	elicit_request/5	1916
4.447	empty/1	1916
4.448	enable/0	1916
4.449	enable/1	1916
4.450	enable/2	1917

4.451	enable_logging/1	1917
4.452	enabled/1	1917
4.453	enabled/2	1917
4.454	encode/2	1917
4.455	encode_frame/2	1917
4.456	encoding_suffix/2	1917
4.457	ensure_directory/1	1917
4.458	ensure_file/1	1918
4.459	entity/1	1918
4.460	entity/2	1918
4.461	entity_info_pair_score_hook/3	1918
4.462	entity_info_score_hook/2	1918
4.463	entity_mutants/2	1918
4.464	entity_mutants/3	1918
4.465	entity_predicates_weights_hook/2	1919
4.466	entity_prefix/2	1919
4.467	entity_score/2	1919
4.468	enumerate/2	1919
4.469	environment_variable/2	1919
4.470	epsilon/1	1919
4.471	equal/2	1919
4.472	equirectangular_inverse/4	1919
4.473	equirectangular_projection/4	1920
4.474	erase/1	1920
4.475	error/2	1920
4.476	error_code/2	1920
4.477	error_data/2	1920
4.478	error_message/2	1920
4.479	error_response/4	1920
4.480	error_response/5	1920
4.481	essentially_equal/3	1921
4.482	estimate_temperature/1	1921
4.483	estimate_temperature/2	1921
4.484	euclidean_distance/3	1921
4.485	euclidean_norm/2	1921
4.486	example/3	1921
4.487	exchange_bind/4	1921
4.488	exchange_declare/3	1921
4.489	exchange_delete/3	1922
4.490	exchange_unbind/4	1922
4.491	exclude/3	1922
4.492	execution_context/7	1922
4.493	exists/3	1922
4.494	expand/2	1922
4.495	expand_library_path/2	1922
4.496	expected/1	1922
4.497	expecteds/2	1923
4.498	expire/4	1923
4.499	explain/2	1923
4.500	explain//1	1923
4.501	exponential/2	1923
4.502	export/1	1923
4.503	export/2	1923
4.504	extension/1	1923

4.505 extension_type/2	1924
4.506 extension_type/3	1924
4.507 extensions/1	1924
4.508 external_reference/2	1924
4.509 factorial/2	1924
4.510 facts/1	1924
4.511 failed_test_reason//1	1924
4.512 false/1	1924
4.513 fcube/0	1925
4.514 file/1	1925
4.515 file/2	1925
4.516 file_exists/1	1925
4.517 file_footer/3	1925
4.518 file_header/3	1925
4.519 file_modification_time/2	1926
4.520 file_path_components/2	1926
4.521 file_permission/2	1926
4.522 file_score/2	1926
4.523 file_size/2	1926
4.524 file_to_bytes/2	1926
4.525 file_to_bytes/3	1926
4.526 file_to_chars/2	1926
4.527 file_to_chars/3	1927
4.528 file_to_codes/2	1927
4.529 file_to_codes/3	1927
4.530 file_to_terms/2	1927
4.531 file_to_terms/3	1927
4.532 file_type_extension/2	1927
4.533 files/1	1927
4.534 files/2	1928
4.535 files/3	1928
4.536 filter/2	1928
4.537 filter/3	1928
4.538 final_bearing/3	1928
4.539 find/4	1928
4.540 find/5	1928
4.541 findall_in_noblock/3	1929
4.542 findall_in_noblock/4	1929
4.543 findall_member/4	1929
4.544 findall_member/5	1929
4.545 findall_rd_noblock/3	1929
4.546 findall_rd_noblock/4	1929
4.547 finished_by/2	1929
4.548 finishes/2	1929
4.549 fisher/3	1930
4.550 flag_group_chk/1	1930
4.551 flag_groups/1	1930
4.552 flat_map/2	1930
4.553 flatten/1	1930
4.554 flatten/2	1930
4.555 float//1	1930
4.556 flush_all/1	1931
4.557 flush_all/2	1931
4.558 fold_left/4	1931

4.559	fold_left/3	1931
4.560	fold_right/4	1931
4.561	fold_right_1/3	1931
4.562	fordownto/3	1931
4.563	fordownto/4	1931
4.564	fordownto/5	1932
4.565	foreach/3	1932
4.566	foreach/4	1932
4.567	format/2	1932
4.568	format/3	1932
4.569	format_date_time/4	1932
4.570	format_entity_score//2	1932
4.571	format_object/1	1932
4.572	format_report/1	1933
4.573	format_report/2	1933
4.574	format_report/3	1933
4.575	format_to_atom/3	1933
4.576	format_to_chars/3	1933
4.577	format_to_chars/4	1933
4.578	format_to_codes/3	1933
4.579	format_to_codes/4	1933
4.580	forto/3	1934
4.581	forto/4	1934
4.582	forto/5	1934
4.583	forward/1	1934
4.584	forward/2	1934
4.585	forward/3	1934
4.586	fractile/3	1934
4.587	frame_body/2	1934
4.588	frame_command/2	1935
4.589	frame_header/3	1935
4.590	frame_headers/2	1935
4.591	freeze/2	1935
4.592	frequency_distribution/2	1935
4.593	from_expected/2	1935
4.594	from_generator/2	1935
4.595	from_generator/3	1935
4.596	from_generator/4	1936
4.597	from_goal/2	1936
4.598	from_goal/3	1936
4.599	from_goal/4	1936
4.600	from_goal_or_throw/2	1936
4.601	from_goal_or_throw/3	1936
4.602	from_optional/3	1936
4.603	frozen/2	1937
4.604	full_device_path/1	1937
4.605	func_test/3	1937
4.606	functional/0	1937
4.607	gamma/3	1937
4.608	gat/4	1937
4.609	gats/5	1937
4.610	generate/1	1937
4.611	generate/2	1938
4.612	generate/3	1938

4.613	generate/4	1938
4.614	generate/8	1939
4.615	generate_all/2	1939
4.616	genint/2	1939
4.617	gensym/2	1939
4.618	geometric/2	1939
4.619	geometric_mean/2	1939
4.620	get/1	1939
4.621	get/3	1939
4.622	get/4	1940
4.623	get_field/2	1940
4.624	get_flag_value/2	1940
4.625	get_seed/1	1940
4.626	getrange/5	1940
4.627	gets/4	1940
4.628	gets/5	1940
4.629	git_object_identifier/1	1940
4.630	gnu/0	1941
4.631	goal_expansion/2	1941
4.632	graph_coloring/3	1941
4.633	graph_footer/5	1941
4.634	graph_header/5	1941
4.635	ground/1	1941
4.636	group_by_key/2	1941
4.637	group_consecutive_by_key/2	1941
4.638	group_sorted_by_key/2	1942
4.639	guess_all_extensions/2	1942
4.640	guess_all_extensions/3	1942
4.641	guess_arity/2	1942
4.642	guess_extension/2	1942
4.643	guess_extension/3	1942
4.644	guess_file_type/3	1942
4.645	guess_file_type/4	1942
4.646	guess_separator/2	1943
4.647	guess_type/3	1943
4.648	guess_type/4	1943
4.649	gumbel/3	1943
4.650	hamming/3	1943
4.651	hamming_distance/3	1943
4.652	handbook/0	1943
4.653	harmonic_mean/2	1943
4.654	has_cycle/1	1944
4.655	has_negative_cycle/1	1944
4.656	has_path/3	1944
4.657	hash/2	1944
4.658	haversine_distance/3	1944
4.659	hdel/4	1944
4.660	head/2	1944
4.661	head_pred/1	1944
4.662	help/0	1945
4.663	help/1	1945
4.664	help/2	1945
4.665	help/3	1945
4.666	hex_digit/1	1945

4.667 hex_digits//1	1945
4.668 hexists/4	1945
4.669 hget/4	1945
4.670 hgetall/3	1946
4.671 hkeys/3	1946
4.672 hlen/3	1946
4.673 home/1	1946
4.674 homepage/1	1946
4.675 hset/5	1946
4.676 hvals/3	1946
4.677 hypergeometric/4	1946
4.678 ibk/3	1947
4.679 id/2	1947
4.680 if_empty/1	1947
4.681 if_expected/1	1947
4.682 if_expected_or_else/2	1947
4.683 if_invalid/1	1947
4.684 if_present/1	1947
4.685 if_present_or_else/2	1947
4.686 if_unexpected/1	1948
4.687 if_valid/1	1948
4.688 if_valid_or_else/2	1948
4.689 in/1	1948
4.690 in/2	1948
4.691 in_degree/3	1948
4.692 in_list/2	1948
4.693 in_list/3	1948
4.694 in_noblock/1	1949
4.695 in_noblock/2	1949
4.696 include/3	1949
4.697 incr/3	1949
4.698 incr/4	1949
4.699 incrby/4	1949
4.700 increase/1	1949
4.701 increment/0	1949
4.702 increment_counter/1	1950
4.703 init/0	1950
4.704 init/2	1950
4.705 init1/2	1950
4.706 init_log_file/2	1950
4.707 init_tail/2	1950
4.708 init_tails/2	1950
4.709 initial_bearing/3	1950
4.710 initial_state/1	1951
4.711 initial_temperature/1	1951
4.712 initialize/1	1951
4.713 inits/2	1951
4.714 inits1/2	1951
4.715 inorder/2	1951
4.716 insert/3	1951
4.717 insert/4	1951
4.718 insert_after/3	1952
4.719 insert_all/3	1952
4.720 insert_before/3	1952

4.721	insert_in/4	1952
4.722	install/1	1952
4.723	install/2	1952
4.724	install/3	1952
4.725	install/4	1952
4.726	installed/0	1953
4.727	installed/1	1953
4.728	installed/3	1953
4.729	installed/4	1953
4.730	instance/1	1953
4.731	instance/2	1953
4.732	instances/1	1953
4.733	instant_to_utc_date_time/2	1953
4.734	integer//1	1954
4.735	integer_to_big_endian_bytes32/2	1954
4.736	integer_to_big_endian_bytes64/2	1954
4.737	integer_to_little_endian_bytes32/2	1954
4.738	internal_error/2	1954
4.739	internal_os_path/2	1954
4.740	interpolate_great_circle/4	1954
4.741	interpolate_rhumb/4	1954
4.742	interquartile_range/2	1955
4.743	intersect/2	1955
4.744	intersection/2	1955
4.745	intersection/3	1955
4.746	intersection/4	1955
4.747	interval_string/2	1955
4.748	invalid/1	1955
4.749	invalid_params/2	1955
4.750	invalid_request/1	1956
4.751	invalids/2	1956
4.752	invoke/1	1956
4.753	invoke/2	1956
4.754	ipv4//1	1956
4.755	ipv6//1	1956
4.756	is_absolute_file_name/1	1956
4.757	is_acyclic/1	1956
4.758	is_alpha/1	1957
4.759	is_alphanumeric/1	1957
4.760	is_ascii/1	1957
4.761	is_batch/1	1957
4.762	is_bin_digit/1	1957
4.763	is_bipartite/1	1957
4.764	is_clockwise_polygon/1	1957
4.765	is_complete/1	1957
4.766	is_connected/1	1958
4.767	is_control/1	1958
4.768	is_dec_digit/1	1958
4.769	is_empty/0	1958
4.770	is_end_of_line/1	1958
4.771	is_error_response/1	1958
4.772	is_expected/0	1958
4.773	is_false/1	1958
4.774	is_hex_digit/1	1959

4.775	is_invalid/0	1959
4.776	is_layout/1	1959
4.777	is_letter/1	1959
4.778	is_lower_case/1	1959
4.779	is_newline/1	1959
4.780	is_notification/1	1959
4.781	is_null/1	1959
4.782	is_object/1	1960
4.783	is_octal_digit/1	1960
4.784	is_period/1	1960
4.785	is_prefix_of/2	1960
4.786	is_present/0	1960
4.787	is_punctuation/1	1960
4.788	is_quote/1	1960
4.789	is_request/1	1960
4.790	is_response/1	1961
4.791	is_sparse/1	1961
4.792	is_subsequence_of/2	1961
4.793	is_suffix_of/2	1961
4.794	is_tree/1	1961
4.795	is_true/1	1961
4.796	is_unexpected/0	1961
4.797	is_upper_case/1	1961
4.798	is_valid/0	1962
4.799	is_valid_polygon/1	1962
4.800	is_void/1	1962
4.801	is_vowel/1	1962
4.802	is_white_space/1	1962
4.803	iterator_element/2	1962
4.804	jaccard_index/3	1962
4.805	jaro/3	1962
4.806	jaro_winkler/3	1963
4.807	join/3	1963
4.808	join_all/3	1963
4.809	jump/3	1963
4.810	jump_all/3	1963
4.811	jump_all_block/3	1963
4.812	k_distinct_subsequence/3	1963
4.813	k_distinct_subsequences/3	1963
4.814	k_permutation/3	1964
4.815	k_permutation/4	1964
4.816	k_permutations/3	1964
4.817	k_permutations/4	1964
4.818	key/2	1964
4.819	keys/2	1964
4.820	keys/3	1964
4.821	keys_values/3	1964
4.822	keysort/2	1965
4.823	kill/1	1965
4.824	kill/2	1965
4.825	known_zone_id/1	1965
4.826	kurtosis/2	1965
4.827	language_object/2	1965
4.828	last/2	1965

4.829	leaf/1	1965
4.830	leaf_class/1	1966
4.831	leaf_classes/1	1966
4.832	leaf_instance/1	1966
4.833	leaf_instances/1	1966
4.834	leap_effective_date/2	1966
4.835	leap_offset_at_utc_unix/2	1966
4.836	leap_second_date/2	1966
4.837	leap_seconds_entries/1	1966
4.838	leap_seconds_source/1	1967
4.839	leap_year/1	1967
4.840	learn/0	1967
4.841	learn/1	1967
4.842	learn/2	1967
4.843	learn/3	1967
4.844	learn_seq/2	1967
4.845	learn_with_timeout/4	1968
4.846	leash/1	1968
4.847	leashing/1	1968
4.848	least_common_multiple/2	1968
4.849	leaves/1	1968
4.850	leaves/2	1968
4.851	length/2	1968
4.852	levenshtein/3	1969
4.853	libraries/0	1969
4.854	libraries/1	1969
4.855	libraries/2	1969
4.856	libraries/3	1969
4.857	library/1	1969
4.858	library/2	1970
4.859	library_mutants/2	1970
4.860	library_mutants/3	1970
4.861	library_score/2	1970
4.862	license/1	1970
4.863	linda/0	1970
4.864	linda/1	1970
4.865	linda_client/1	1971
4.866	linda_client/2	1971
4.867	linda_timeout/2	1971
4.868	line_to_chars/2	1971
4.869	line_to_chars/3	1971
4.870	line_to_codes/2	1971
4.871	line_to_codes/3	1971
4.872	linear_regression/4	1971
4.873	lint/0	1972
4.874	lint/1	1972
4.875	lint/2	1972
4.876	list/0	1972
4.877	list_to_array/2	1972
4.878	listing/0	1972
4.879	listing/1	1972
4.880	little_endian_word32/2	1972
4.881	llen/3	1973
4.882	load/1	1973

4.883	load/2	1973
4.884	load_config/1	1973
4.885	load_coverage_config/1	1973
4.886	load_dut1_override/1	1973
4.887	load_leap_seconds_override/1	1973
4.888	load_program/1	1974
4.889	loaded_file/1	1974
4.890	loaded_file_property/2	1974
4.891	loaded_files_topological_sort/1	1974
4.892	loaded_files_topological_sort/2	1974
4.893	loaded_pack/3	1974
4.894	loaded_pack_dependency/6	1974
4.895	loaded_pack_file/4	1974
4.896	loader_file/1	1975
4.897	local_abbreviation/2	1975
4.898	local_abbreviation/3	1975
4.899	local_abbreviation/4	1975
4.900	local_abbreviation_reified/2	1975
4.901	local_abbreviation_reified/3	1975
4.902	local_abbreviation_reified/4	1975
4.903	local_abbreviation_with_resolution/3	1975
4.904	local_abbreviation_with_resolution/4	1976
4.905	local_abbreviation_with_resolution/5	1976
4.906	local_daylight_saving_time/2	1976
4.907	local_daylight_saving_time/3	1976
4.908	local_daylight_saving_time/4	1976
4.909	local_daylight_saving_time_reified/2	1976
4.910	local_daylight_saving_time_reified/3	1976
4.911	local_daylight_saving_time_reified/4	1976
4.912	local_daylight_saving_time_with_resolution/3	1977
4.913	local_daylight_saving_time_with_resolution/4	1977
4.914	local_daylight_saving_time_with_resolution/5	1977
4.915	local_offset/2	1977
4.916	local_offset/3	1977
4.917	local_offset/4	1977
4.918	local_offset_reified/2	1977
4.919	local_offset_reified/3	1977
4.920	local_offset_reified/4	1978
4.921	local_offset_with_resolution/3	1978
4.922	local_offset_with_resolution/4	1978
4.923	local_offset_with_resolution/5	1978
4.924	local_time_type/2	1978
4.925	local_time_type/3	1978
4.926	local_time_type/4	1978
4.927	local_time_type_reified/2	1978
4.928	local_time_type_reified/3	1979
4.929	local_time_type_reified/4	1979
4.930	local_time_type_with_resolution/3	1979
4.931	local_time_type_with_resolution/4	1979
4.932	local_time_type_with_resolution/5	1979
4.933	local_to_utc/3	1979
4.934	local_to_utc_tz/3	1979
4.935	local_to_utc_tz_with_resolution/4	1979
4.936	log/3	1980

4.937	log_event/2	1980
4.938	log_file/2	1980
4.939	logging/1	1980
4.940	logging/3	1980
4.941	logistic/3	1980
4.942	lognormal/3	1980
4.943	logseries/2	1980
4.944	logtalk_packs/0	1981
4.945	logtalk_packs/1	1981
4.946	long_flags/1	1981
4.947	longest_common_increasing_subsequence/3	1981
4.948	longest_common_subsequence/3	1981
4.949	longest_common_subsequence_length/3	1981
4.950	longest_common_substring/3	1981
4.951	longest_decreasing_subsequence/2	1981
4.952	longest_increasing_subsequence/2	1982
4.953	longest_repeating_subsequence/2	1982
4.954	lookup/2	1982
4.955	lookup/3	1982
4.956	lookup/4	1982
4.957	lookup_in/3	1982
4.958	lower_upper/2	1982
4.959	lpop/3	1982
4.960	lpush/4	1983
4.961	lrange/5	1983
4.962	lrem/5	1983
4.963	ltrim/5	1983
4.964	magic/2	1983
4.965	magicise/4	1983
4.966	make_directory/1	1983
4.967	make_directory_path/1	1983
4.968	make_set/3	1984
4.969	man/1	1984
4.970	manhattan_distance/3	1984
4.971	manhattan_norm/2	1984
4.972	map/2	1984
4.973	map/3	1984
4.974	map/4	1985
4.975	map/5	1985
4.976	map/6	1985
4.977	map/7	1985
4.978	map/8	1985
4.979	map_both/3	1985
4.980	map_catching/2	1985
4.981	map_element/2	1985
4.982	map_invalid/2	1986
4.983	map_or_else/3	1986
4.984	map_reduce/5	1986
4.985	map_unexpected/2	1986
4.986	mask32/1	1986
4.987	mask64/1	1986
4.988	materialize/0	1986
4.989	max/2	1986
4.990	max/3	1987

4.991	max_clauses/1	1987
4.992	max_inv_preds/1	1987
4.993	max_path/5	1987
4.994	max_size/1	1987
4.995	max_tree/3	1987
4.996	maximal_cliques/2	1987
4.997	maximum_cliques/2	1987
4.998	maybe/0	1988
4.999	maybe/1	1988
4.1000	maybe/2	1988
4.1001	maybe_call/1	1988
4.1002	maybe_call/2	1988
4.1003	mean_center/2	1988
4.1004	mean_deviation/2	1988
4.1005	mean_squared_error/3	1988
4.1006	median/2	1989
4.1007	median_deviation/2	1989
4.1008	meets/2	1989
4.1009	member/2	1989
4.1010	memberchk/2	1989
4.1011	merge/3	1989
4.1012	message_body/2	1989
4.1013	message_delivery_tag/2	1990
4.1014	message_exchange/2	1990
4.1015	message_hook/4	1990
4.1016	message_prefix_file/6	1990
4.1017	message_prefix_stream/4	1990
4.1018	message_properties/2	1990
4.1019	message_property/3	1990
4.1020	message_routing_key/2	1990
4.1021	message_tokens//2	1991
4.1022	met_by/2	1991
4.1023	meta/1	1991
4.1024	meta_type/3	1991
4.1025	metaphone/2	1991
4.1026	metaphone_match/2	1991
4.1027	metarule/6	1991
4.1028	metarule_next_id/1	1991
4.1029	method/2	1992
4.1030	method_not_found/2	1992
4.1031	lmgget/3	1992
4.1032	mibenum/1	1992
4.1033	midpoint/3	1992
4.1034	min/2	1992
4.1035	min/3	1992
4.1036	min_clauses/1	1993
4.1037	min_distances/3	1993
4.1038	min_max/3	1993
4.1039	min_max_normalization/2	1993
4.1040	min_path/5	1993
4.1041	min_path_bellman_ford/5	1993
4.1042	nin_paths/3	1993
4.1043	nin_predecessors/3	1993
4.1044	min_tree/3	1994

4.1045	minimum_enclosing_circle/3	1994
4.1046	modes/2	1994
4.1047	module_property/2	1994
4.1048	monitor/1	1994
4.1049	monitor/4	1994
4.1050	monitor_activated/0	1994
4.1051	monitored/1	1994
4.1052	monitors/1	1995
4.1053	month_weekday_date/5	1995
4.1054	mset/3	1995
4.1055	msort/2	1995
4.1056	msort/3	1995
4.1057	mul32/3	1995
4.1058	mul64/3	1995
4.1059	mutation/2	1995
4.1060	mutation/3	1996
4.1061	nack/3	1996
4.1062	name/1	1996
4.1063	name_of_day/3	1996
4.1064	name_of_month/3	1996
4.1065	natural//1	1996
4.1066	nearest_coordinate/5	1996
4.1067	nearest_point_on_polyline/4	1997
4.1068	nearest_point_on_segment/4	1997
4.1069	neighbor_state/2	1997
4.1070	neighbor_state/3	1997
4.1071	neighbors/3	1997
4.1072	new/1	1997
4.1073	new/2	1997
4.1074	new/3	1998
4.1075	new_line//0	1998
4.1076	new_lines//0	1998
4.1077	next/2	1998
4.1078	next/3	1998
4.1079	next/4	1998
4.1080	next_permutation/2	1998
4.1081	nextto/3	1998
4.1082	node/7	1999
4.1083	nodebug/0	1999
4.1084	nolog/3	1999
4.1085	nologall/0	1999
4.1086	non_blank//1	1999
4.1087	non_blanks//1	1999
4.1088	nonempty_subsequence/2	1999
4.1089	nonempty_subsequences/2	1999
4.1090	normal/3	2000
4.1091	normal_element/2	2000
4.1092	normalize/2	2000
4.1093	normalize_coordinate/2	2000
4.1094	normalize_date_time/2	2000
4.1095	normalize_polygon_orientation/3	2000
4.1096	normalize_range/2	2000
4.1097	normalize_range/4	2000
4.1098	normalize_scalar/2	2001

4.1099normalize_unit/2	2001
4.1100nospy/1	2001
4.1101nospy/3	2001
4.1102nospy/4	2001
4.1103nospyall/0	2001
4.1104not64/2	2001
4.1105note/2	2001
4.1106note/3	2002
4.1107notification/2	2002
4.1108notification/3	2002
4.1109notrace/0	2002
4.1110now/3	2002
4.1111nth0/3	2002
4.1112nth0/4	2002
4.1113nth1/3	2002
4.1114nth1/4	2003
4.1115nth_combination/4	2003
4.1116nth_permutation/3	2003
4.1117null/1	2003
4.1118null_device_path/1	2003
4.1119number//1	2003
4.1120number_of_edges/2	2003
4.1121number_of_tests/1	2003
4.1122number_of_vertices/2	2004
4.1123number_string/2	2004
4.1124numbervars/1	2004
4.1125numbervars/3	2004
4.1126occurrences/2	2004
4.1127occurrences/3	2004
4.1128occurs/2	2004
4.1129of/2	2004
4.1130of_expected/2	2005
4.1131of_invalid/2	2005
4.1132of_invalids/2	2005
4.1133of_unexpected/2	2005
4.1134of_valid/2	2005
4.1135offset/2	2005
4.1136offset/3	2005
4.1137offset/4	2005
4.1138one_or_more//0	2006
4.1139one_or_more//1	2006
4.1140one_or_more//2	2006
4.1141operating_system_machine/1	2006
4.1142operating_system_name/1	2006
4.1143operating_system_release/1	2006
4.1144operating_system_type/1	2006
4.1145option/2	2006
4.1146option/3	2007
4.1147or/2	2007
4.1148or64/3	2007
4.1149or_else/2	2007
4.1150or_else_call/2	2007
4.1151or_else_fail/1	2007
4.1152or_else_get/2	2008

4.1153	br_else_throw/1	2008
4.1154	br_else_throw/2	2008
4.1155	originator/1	2008
4.1156	orphaned/0	2008
4.1157	orphaned/2	2008
4.1158	out/1	2008
4.1159	out/2	2009
4.1160	out_degree/3	2009
4.1161	outdated/0	2009
4.1162	outdated/1	2009
4.1163	outdated/2	2009
4.1164	outdated/4	2009
4.1165	outdated/5	2009
4.1166	output_file_name/2	2009
4.1167	output_schema/2	2010
4.1168	overlapped_by/2	2010
4.1169	overlaps/2	2010
4.1170	pack_dependency/6	2010
4.1171	pack_metadata/4	2010
4.1172	pack_object/3	2010
4.1173	pack_property/4	2010
4.1174	package/1	2010
4.1175	pad_md/4	2011
4.1176	params/2	2011
4.1177	parent/1	2011
4.1178	parenthesis/2	2011
4.1179	parents/1	2011
4.1180	parse/2	2011
4.1181	parse/3	2012
4.1182	parse/4	2012
4.1183	parse/5	2012
4.1184	parse_all/2	2012
4.1185	parse_domain/2	2012
4.1186	parse_domain/3	2012
4.1187	parse_error/1	2012
4.1188	parse_problem/2	2013
4.1189	parse_problem/3	2013
4.1190	partial_map/4	2013
4.1191	partition/3	2013
4.1192	partition/4	2013
4.1193	partition/5	2013
4.1194	partition/6	2013
4.1195	path/3	2013
4.1196	path_concat/3	2014
4.1197	peek_back/2	2014
4.1198	peek_front/2	2014
4.1199	percentile/3	2014
4.1200	permutation/2	2014
4.1201	permutation/3	2014
4.1202	permutation_index/3	2014
4.1203	permutations/2	2015
4.1204	permutations/3	2015
4.1205	persist/3	2015
4.1206	pid/1	2015

4.1207	pin/0	2015
4.1208	pin/1	2015
4.1209	pinned/1	2015
4.1210	plus/3	2015
4.1211	point_in_polygon/2	2016
4.1212	point_to_polyline_distance/3	2016
4.1213	poisson/2	2016
4.1214	polygon_area/2	2016
4.1215	polygon_bounding_box/2	2016
4.1216	polygon_centroid/2	2016
4.1217	polygon_orientation/2	2016
4.1218	polygon_perimeter/2	2016
4.1219	polygon_perimeter/3	2017
4.1220	polygons_intersect/2	2017
4.1221	polyline_length/2	2017
4.1222	polyline_length/3	2017
4.1223	polyline_resample/3	2017
4.1224	polyline_simplify/3	2017
4.1225	polyline_split_at_distance/4	2017
4.1226	pop_back/3	2017
4.1227	pop_front/3	2018
4.1228	port/5	2018
4.1229	portray_clause/1	2018
4.1230	postorder/2	2018
4.1231	power/2	2018
4.1232	power_sequence/4	2018
4.1233	power_set/2	2018
4.1234	powerset/2	2018
4.1235	pp/1	2019
4.1236	pprint/1	2019
4.1237	predicate/1	2019
4.1238	predicate/2	2019
4.1239	predicate/3	2019
4.1240	predicate_info_pair_score_hook/4	2019
4.1241	predicate_info_score_hook/3	2019
4.1242	predicate_mode_score_hook/3	2020
4.1243	predicate_mode_score_hook/5	2020
4.1244	predicate_mutants/3	2020
4.1245	predicate_mutants/4	2020
4.1246	predicate_stratum/3	2020
4.1247	predicates/2	2020
4.1248	predicates/3	2020
4.1249	predict/3	2020
4.1250	predict/4	2021
4.1251	predict_probabilities/3	2021
4.1252	predict_probabilities/4	2021
4.1253	preferred_mime_name/1	2021
4.1254	prefix/0	2021
4.1255	prefix/1	2021
4.1256	prefix/2	2022
4.1257	prefix/3	2022
4.1258	preorder/2	2022
4.1259	prepend/3	2022
4.1260	previous/2	2022

4.1261	previous/3	2022
4.1262	previous/4	2022
4.1263	previous_permutation/2	2022
4.1264	print_classifier/1	2023
4.1265	print_flags/0	2023
4.1266	print_flags/1	2023
4.1267	print_message/3	2023
4.1268	print_message_token/4	2023
4.1269	print_message_tokens/3	2023
4.1270	product/2	2023
4.1271	product/3	2023
4.1272	program_to_clauses/2	2024
4.1273	progress/5	2024
4.1274	prompt_get/3	2024
4.1275	prompts/1	2024
4.1276	proper_prefix/2	2024
4.1277	proper_prefix/3	2024
4.1278	proper_subsequence/2	2024
4.1279	proper_suffix/2	2024
4.1280	proper_suffix/3	2025
4.1281	property/1	2025
4.1282	prove/2	2025
4.1283	prove/3	2025
4.1284	provides/2	2025
4.1285	prune/3	2025
4.1286	prune/5	2025
4.1287	push_back/3	2025
4.1288	push_front/3	2026
4.1289	quartiles/4	2026
4.1290	query/1	2026
4.1291	query/2	2026
4.1292	question_hook/6	2026
4.1293	question_prompt_stream/4	2026
4.1294	queue_bind/4	2026
4.1295	queue_declare/3	2026
4.1296	queue_delete/3	2027
4.1297	queue_purge/2	2027
4.1298	queue_unbind/4	2027
4.1299	quick_check/1	2027
4.1300	quick_check/2	2027
4.1301	quick_check/3	2027
4.1302	random/1	2027
4.1303	random/3	2027
4.1304	random_combination/3	2028
4.1305	random_node/1	2028
4.1306	random_permutation/2	2028
4.1307	random_subsequence/2	2028
4.1308	random_tree/1	2028
4.1309	randomize/1	2028
4.1310	randseq/4	2028
4.1311	randset/4	2028
4.1312	range/2	2029
4.1313	rank_correlation/3	2029
4.1314	rd/1	2029

4.1315rd/2	2029
4.1316rd_list/2	2029
4.1317rd_list/3	2029
4.1318d_noblock/1	2029
4.1319d_noblock/2	2029
4.1320directories/1	2030
4.1321directories/2	2030
4.1322directory/1	2030
4.1323directory/2	2030
4.1324directory/3	2030
4.1325directory_score/2	2031
4.1326reachable/3	2031
4.1327read_file/2	2031
4.1328read_file/3	2031
4.1329read_file_by_line/2	2031
4.1330read_file_by_line/3	2031
4.1331read_framed_message/2	2031
4.1332read_from_atom/2	2032
4.1333read_from_chars/2	2032
4.1334read_from_codes/2	2032
4.1335read_message/2	2032
4.1336read_mime_types/2	2032
4.1337read_only_device_path/1	2032
4.1338read_stream/2	2032
4.1339read_stream/3	2032
4.1340read_stream_by_line/2	2033
4.1341read_stream_by_line/3	2033
4.1342read_term_from_atom/3	2033
4.1343read_term_from_chars/3	2033
4.1344read_term_from_chars/4	2033
4.1345read_term_from_codes/3	2033
4.1346read_term_from_codes/4	2033
4.1347readme/1	2033
4.1348readme/2	2034
4.1349receive/3	2034
4.1350recorda/2	2034
4.1351recorda/3	2034
4.1352recorded/2	2034
4.1353recorded/3	2034
4.1354recordz/2	2034
4.1355recordz/3	2034
4.1356relative_standard_deviation/2	2035
4.1357release/1	2035
4.1358release_date/1	2035
4.1359removeDependent/1	2035
4.1360remove_duplicates/2	2035
4.1361remove_rule/1	2035
4.1362rename/4	2035
4.1363rename_file/2	2035
4.1364replace/3	2036
4.1365replace/5	2036
4.1366replace_sub_atom/4	2036
4.1367report_directory/3	2036
4.1368report_entity/3	2036

4.136	report_library/3	2036
4.137	report_predicate/4	2036
4.137	repository/1	2036
4.137	repository_branch/1	2037
4.137	repository_commit/1	2037
4.137	repository_commit_abbreviated/1	2037
4.137	repository_commit_author/1	2037
4.137	repository_commit_date/1	2037
4.137	repository_commit_message/1	2037
4.137	request/3	2037
4.137	request/4	2037
4.138	rescale/3	2038
4.138	reset/0	2038
4.138	reset/1	2038
4.138	reset_counter/1	2038
4.138	reset_counters/0	2038
4.138	reset_flags/0	2038
4.138	reset_flags/1	2038
4.138	reset_genint/0	2039
4.138	reset_genint/1	2039
4.138	reset_gensym/0	2039
4.139	reset_gensym/1	2039
4.139	reset_monitor/0	2039
4.139	reset_seed/0	2039
4.139	resize/2	2039
4.139	resource_read/3	2039
4.139	resources/1	2040
4.139	response/3	2040
4.139	restore/1	2040
4.139	restore/2	2040
4.139	result/2	2040
4.140	retract_fact/1	2040
4.140	reverse/2	2040
4.140	rewind/2	2040
4.140	rewind/3	2041
4.140	rhumb_bearing/3	2041
4.140	rhumb_destination_point/4	2041
4.140	rhumb_distance/3	2041
4.140	rhumb_midpoint/3	2041
4.140	libraries/1	2041
4.140	libraries/2	2041
4.141	library/1	2041
4.141	library/2	2042
4.141	library_score/2	2042
4.141	ol32/3	2042
4.141	ol64/3	2042
4.141	rollback/0	2042
4.141	root_mean_squared_error/3	2042
4.141	ror32/3	2042
4.141	route_distance/2	2043
4.141	route_distance/3	2043
4.142	route_distance/4	2043
4.142	rpop/3	2043
4.142	push/4	2043

4.1423	rule/2	2043
4.1424	rule/3	2043
4.1425	rule/4	2043
4.1426	rules/1	2044
4.1427	run/0	2044
4.1428	run/1	2044
4.1429	run/2	2044
4.1430	run/3	2044
4.1431	run/4	2044
4.1432	run_test_sets/1	2044
4.1433	sadd/4	2044
4.1434	same_instant/2	2045
4.1435	same_length/2	2045
4.1436	same_length/3	2045
4.1437	save/0	2045
4.1438	save/1	2045
4.1439	save/2	2045
4.1440	save_dut1_entries/1	2045
4.1441	save_leap_seconds_entries/1	2046
4.1442	scalar_product/3	2046
4.1443	scan_left/4	2046
4.1444	scan_left_1/3	2046
4.1445	scan_right/4	2046
4.1446	scan_right_1/3	2046
4.1447	scard/3	2046
4.1448	score/3	2046
4.1449	score_all/3	2047
4.1450	search/1	2047
4.1451	secondary_header/2	2047
4.1452	secondary_header_flag/2	2047
4.1453	secondary_header_time/2	2047
4.1454	select/3	2047
4.1455	select/4	2047
4.1456	selectchk/3	2048
4.1457	selectchk/4	2048
4.1458	send/3	2048
4.1459	send/4	2048
4.1460	send_heartbeat/1	2048
4.1461	sequence/2	2048
4.1462	sequence/3	2048
4.1463	sequence/4	2049
4.1464	sequence/5	2049
4.1465	sequence_count/2	2049
4.1466	sequence_flags/2	2049
4.1467	sequential_occurrences/2	2049
4.1468	sequential_occurrences/3	2049
4.1469	serve/3	2049
4.1470	server_accept/4	2049
4.1471	server_accept/5	2050
4.1472	server_close/1	2050
4.1473	server_open/2	2050
4.1474	server_open/3	2050
4.1475	server_open/4	2050
4.1476	set/1	2050

4.147	set/3	2050
4.147	set/4	2050
4.147	set/5	2051
4.148	set_element/2	2051
4.148	set_field/2	2051
4.148	set_flag_value/2	2051
4.148	set_flag_value/3	2051
4.148	set_monitor/4	2051
4.148	set_seed/1	2051
4.148	set_spy_point/4	2051
4.148	set_write_max_depth/1	2052
4.148	setrange/5	2052
4.148	setup/0	2052
4.149	shell/1	2052
4.149	shell/2	2052
4.149	shell_command/1	2052
4.149	shl64/3	2052
4.149	short_flags/1	2052
4.149	shr64/3	2053
4.149	shrink/3	2053
4.149	shrink_sequence/3	2053
4.149	shrinker/1	2053
4.149	shutdown_server/1	2053
4.150	sign//1	2053
4.150	singletons/2	2053
4.150	ismember/4	2053
4.150	size/2	2054
4.150	skewness/2	2054
4.150	sleep/1	2054
4.150	sliding_window/3	2054
4.150	smembers/3	2054
4.150	softmax/2	2054
4.150	softmax/3	2054
4.151	software_heritage_identifier/1	2054
4.151	sort/2	2055
4.151	sort/3	2055
4.151	sort/4	2055
4.151	soundex/2	2055
4.151	soundex_match/2	2055
4.151	source_file_extension/1	2055
4.151	space//0	2055
4.151	spaces//0	2055
4.151	split/3	2056
4.152	split/4	2056
4.152	split_string/4	2056
4.152	spy/1	2056
4.152	spy/3	2056
4.152	spy/4	2056
4.152	spy_point/4	2056
4.152	spying/1	2056
4.152	spying/3	2057
4.152	spying/4	2057
4.152	rem/4	2057
4.153	standard_cauchy/3	2057

4.1531	standard_deviation/2	2057
4.1532	standard_error/2	2057
4.1533	standard_exponential/1	2057
4.1534	standard_gamma/2	2057
4.1535	standard_normal/1	2058
4.1536	standard_t/2	2058
4.1537	start/0	2058
4.1538	start/2	2058
4.1539	start/3	2058
4.1540	start/4	2058
4.1541	start/5	2058
4.1542	start_redirect_to_file/2	2058
4.1543	started_by/2	2059
4.1544	starts/2	2059
4.1545	state_energy/2	2059
4.1546	stats/1	2059
4.1547	stats/2	2059
4.1548	stats/3	2059
4.1549	stem/2	2059
4.1550	stems/2	2059
4.1551	stop/0	2060
4.1552	stop_condition/3	2060
4.1553	stop_redirect_to_file/0	2060
4.1554	strata/1	2060
4.1555	stream_to_bytes/2	2060
4.1556	stream_to_bytes/3	2060
4.1557	stream_to_chars/2	2060
4.1558	stream_to_chars/3	2060
4.1559	stream_to_codes/2	2061
4.1560	stream_to_codes/3	2061
4.1561	stream_to_terms/2	2061
4.1562	stream_to_terms/3	2061
4.1563	string_chars/2	2061
4.1564	string_codes/2	2061
4.1565	string_concat/3	2061
4.1566	string_length/2	2061
4.1567	string_lower/2	2062
4.1568	string_upper/2	2062
4.1569	strlen/3	2062
4.1570	strongly_connected_components/2	2062
4.1571	sub_string/5	2062
4.1572	subclass/1	2062
4.1573	subclasses/1	2062
4.1574	sublist/2	2062
4.1575	subscribe/4	2063
4.1576	subsequence/2	2063
4.1577	subsequence/3	2063
4.1578	subsequence/4	2063
4.1579	subsequence_at_indices/3	2063
4.1580	subsequence_length/2	2063
4.1581	subsequences/2	2063
4.1582	subsequences/3	2063
4.1583	subsequences_with_min_span/3	2064
4.1584	subset/2	2064

4.1585	subslices/2	2064
4.1586	substitute/4	2064
4.1587	subsumes/2	2064
4.1588	subterm/2	2064
4.1589	subtract/3	2064
4.1590	subtract_duration/3	2064
4.1591	succ/2	2065
4.1592	suffix/2	2065
4.1593	suffix/3	2065
4.1594	suffix_alias/2	2065
4.1595	sum/2	2065
4.1596	sum_of_squares/2	2065
4.1597	summary/1	2065
4.1598	superclass/1	2065
4.1599	superclasses/1	2066
4.1600	supplier/1	2066
4.1601	supported_range/2	2066
4.1602	suspend_monitor/0	2066
4.1603	swap/1	2066
4.1604	swap/2	2066
4.1605	swap_consecutive/2	2066
4.1606	syndiff/3	2066
4.1607	symmetric_closure/2	2067
4.1608	tab//0	2067
4.1609	tabs//0	2067
4.1610	tai_minus_utc_for_tai_unix/2	2067
4.1611	tail/2	2067
4.1612	tail1/2	2067
4.1613	tails/2	2067
4.1614	tails1/2	2067
4.1615	take/3	2068
4.1616	take/4	2068
4.1617	tcg_minus_tdb_approx/3	2068
4.1618	tcg_minus_tt_approx/3	2068
4.1619	tdb_minus_tt_approx/3	2068
4.1620	temporary_directory/1	2068
4.1621	term/2	2068
4.1622	term/4	2068
4.1623	term_expansion/2	2069
4.1624	terms_to_array/2	2069
4.1625	test/1	2069
4.1626	time_stamp/1	2069
4.1627	time_string/3	2069
4.1628	time_type/2	2069
4.1629	time_type/3	2069
4.1630	time_type/4	2069
4.1631	timeout/1	2070
4.1632	timestamp/2	2070
4.1633	timestamp/8	2070
4.1634	to_expected/1	2070
4.1635	to_expected/2	2070
4.1636	to_optional/1	2070
4.1637	today/3	2070
4.1638	tolerance_equal/4	2070

4.1639	tool/1	2071
4.1640	tool_call/3	2071
4.1641	tool_call/4	2071
4.1642	tools/0	2071
4.1643	tools/1	2071
4.1644	top/3	2071
4.1645	top_next/5	2071
4.1646	topological_sort/2	2071
4.1647	topological_sort/3	2072
4.1648	touch/3	2072
4.1649	trace/0	2072
4.1650	trace_event/2	2072
4.1651	transitive_closure/2	2072
4.1652	transitive_reduction/2	2072
4.1653	transpose/2	2072
4.1654	traverse/3	2072
4.1655	triangular/4	2073
4.1656	trim/2	2073
4.1657	trim/3	2073
4.1658	trim_left/2	2073
4.1659	trim_left/3	2073
4.1660	trim_right/2	2073
4.1661	trim_right/3	2073
4.1662	trimmed_mean/3	2073
4.1663	true/1	2074
4.1664	t_minus_tai/2	2074
4.1665	ttl/3	2074
4.1666	x_commit/1	2074
4.1667	x_rollback/1	2074
4.1668	x_select/1	2074
4.1669	type/1	2074
4.1670	type/2	2074
4.1671	type/3	2075
4.1672	zdb_version/1	2075
4.1673	unexpected/1	2075
4.1674	unexpecteds/2	2075
4.1675	uniform/1	2075
4.1676	uniform/3	2075
4.1677	uninstall/0	2075
4.1678	uninstall/1	2075
4.1679	uninstall/2	2076
4.1680	union/3	2076
4.1681	union/4	2076
4.1682	union_all/3	2076
4.1683	unix_to_date_time/2	2076
4.1684	unpin/0	2076
4.1685	unpin/1	2076
4.1686	unsubscribe/3	2077
4.1687	unzip/2	2077
4.1688	update/0	2077
4.1689	update/1	2077
4.1690	update/2	2077
4.1691	update/3	2077
4.1692	update/4	2078

4.1693	update/5	2078
4.1694	update_in/4	2078
4.1695	update_in/5	2078
4.1696	user_data/2	2078
4.1697	utc_date_time_to_instant/2	2078
4.1698	utc_to_local/3	2078
4.1699	utc_to_local_tz/3	2078
4.1700	uuid_max/1	2079
4.1701	uuid_nil/1	2079
4.1702	uuid_null/1	2079
4.1703	uuid_v1/2	2079
4.1704	uuid_v3/3	2079
4.1705	uuid_v4/1	2079
4.1706	uuid_v5/3	2079
4.1707	uuid_v7/1	2079
4.1708	valid/0	2080
4.1709	valid/1	2080
4.1710	valid/2	2080
4.1711	valid/3	2080
4.1712	valid_conversion/3	2080
4.1713	valid_coordinate/1	2080
4.1714	valid_date/3	2080
4.1715	valid_date_time/1	2081
4.1716	valid_instant/1	2081
4.1717	valid_option/1	2081
4.1718	valid_options/1	2081
4.1719	valid_scale/1	2081
4.1720	valid_until_date/1	2081
4.1721	validate/1	2081
4.1722	validate/2	2081
4.1723	validate/3	2082
4.1724	valids/2	2082
4.1725	value/1	2082
4.1726	value/3	2082
4.1727	value_reference/2	2082
4.1728	values/2	2082
4.1729	variables/2	2082
4.1730	variance/2	2082
4.1731	variant/2	2083
4.1732	varnumbers/2	2083
4.1733	varnumbers/3	2083
4.1734	verify_commands_availability/0	2083
4.1735	version/1	2083
4.1736	version/2	2083
4.1737	version/6	2083
4.1738	versions/3	2083
4.1739	vertices/2	2084
4.1740	vincenty_distance/3	2084
4.1741	void/1	2084
4.1742	void_element/1	2084
4.1743	von_mises/3	2084
4.1744	wait/2	2084
4.1745	wald/3	2084
4.1746	wall_time/1	2084

4.1747warning/1	2085
4.1748warnings/1	2085
4.1749weakly_connected_components/2	2085
4.1750week_of_year_iso/2	2085
4.1751weekday/2	2085
4.1752weibull/3	2085
4.1753weighted_mean/3	2085
4.1754welcome/0	2085
4.1755when/2	2086
4.1756whiledo/2	2086
4.1757white_space//0	2086
4.1758white_spaces//0	2086
4.1759with_connection/1	2086
4.1760with_output_to/2	2086
4.1761within_distance/4	2086
4.1762without//2	2086
4.1763wneighbors/3	2087
4.1764word32_hex/2	2087
4.1765word64_hex/2	2087
4.1766working_directory/1	2087
4.1767write_file/3	2087
4.1768write_framed_message/2	2087
4.1769write_max_depth/1	2087
4.1770write_message/2	2087
4.1771write_stream/3	2088
4.1772write_term_to_atom/3	2088
4.1773write_term_to_chars/3	2088
4.1774write_term_to_chars/4	2088
4.1775write_term_to_codes/3	2088
4.1776write_term_to_codes/4	2088
4.1777write_to_atom/2	2088
4.1778write_to_chars/2	2088
4.1779write_to_codes/2	2089
4.1780xor64/3	2089
4.1781z_normalization/2	2089
4.1782zadd/5	2089
4.1783zcard/3	2089
4.1784zero_or_more//0	2089
4.1785zero_or_more//1	2089
4.1786zero_or_more//2	2089
4.1787zip/2	2090
4.1788zip/3	2090
4.1789zip_at_index/4	2090
4.1790zone/3	2090
4.1791zone_id_kind/2	2090
4.1792zones/1	2090
4.1793zones/2	2090
4.1794zrange/5	2091
4.1795zrank/4	2091
4.1796zrem/4	2091
4.1797zscore/4	2091

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 ada_boost

object

1.1.1 ada_boost

AdaBoost (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(ada_boost(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-20

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public classifier_protocol
```

Imports:

```
public options
```

Uses:

```
c45
```

`fast_random(Algorithm)`
`format`
`list`
`pairs`
`type`

Remarks:

- Algorithm: AdaBoost iteratively trains weak learners (C4.5 decision trees) on weighted versions of the training data. After each iteration, the weights of misclassified examples are increased so that subsequent learners focus more on difficult cases.
- SAMME variant: This implementation uses the SAMME algorithm (Zhu et al., 2009) which extends AdaBoost to the multi-class case by adjusting the weight update formula to account for the number of classes.
- Learner weights: Each base learner receives a weight (alpha) proportional to its accuracy. More accurate learners have higher weights in the final ensemble vote.
- Classifier representation: The learned classifier is represented as a `ab_classifier(WeightedTrees, ClassValues, Options)` term where `WeightedTrees` is a list of `weighted_tree(Alpha, Tree, AttributeNames)` elements.
- Early stopping: Training stops early if a perfect classifier is found (zero weighted error) or if a base learner performs worse than random guessing.

Inherited public predicates:

`check_option/1` `check_options/1` `classifier_to_clauses/4` `classifier_to_file/4` `default_option/1`
`default_options/1` `learn/2` `option/2` `option/3` `predict/3` `print_classifier/1` `valid_option/1`
`valid_options/1`

- Public predicates
 - `learn/3`
 - `predict_probabilities/3`
- 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`

`predict_probabilities/3`

Predicts class probabilities for a new instance using the learned classifier. Returns a list of Class-Probability pairs sorted by descending probability. Probabilities are derived from the weighted votes of all base learners.

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

`dataset_protocol`, `c45`, `isolation_forest`, `knn`, `naive_bayes`, `nearest_centroid`, `random_forest`

1.2 amqp

object

1.2.1 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: 1:0:0

Date: 2026-02-19

Compilation flags:

`static`, `context_switching_calls`

Uses:

`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:

(none)

- 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
- 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`

`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`

`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`

`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`

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`

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`

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`

`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
```

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
```

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`

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`

`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`

`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`

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`

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`

`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`

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`

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`

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`

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`

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`

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

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

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`

`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`

`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`

`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

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`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

category

1.2.2 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: 1:0:0

Date: 2026-02-19

Compilation flags:

`static`

Uses:

`amqp`

`list`

`os`

Remarks:

(none)

Inherited public predicates:

(none)

- 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

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)

1.3 application

category

1.3.1 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

1.3.2 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)

1.4 arbitrary

category

1.4.1 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: 2:38:0

Date: 2026-02-28

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`.
- 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`

1.5 assertions

object

1.5.1 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

1.5.2 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

1.5.3 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)

1.6 assignvars

object

1.6.1 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:

(\leq)/2 (\Rightarrow)/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

1.6.2 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
 - (\leq)/2
 - (\geq)/2
- Protected predicates
- Private predicates
- Operators
 - op(100,xfx, \leq)
 - op(100,xfx, \geq)

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](#)

1.7 avro

object

1.7.1 avro

Apache Avro binary format parser and generator.

Availability:

```
logtalk_load(avro(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-05

Compilation flags:

```
static, context_switching_calls
```

Uses:

```
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

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

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

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

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.8 base32

object

1.8.1 base32

Base32 encoder and decoder (RFC 4648).

Availability:

`logtalk__load(base32(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2025-01-23

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`

`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`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.9 base58

object

1.9.1 base58

Base58 encoder and decoder (Bitcoin alphabet variant).

Availability:

`logtalk_load(base58(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2025-01-23

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`

`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`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.10 base64

object

1.10.1 base64

Base64 parser and generator.

Availability:

`logtalk__load(base64(loader))`

Author: Paulo Moura

Version: 0:10:0

Date: 2021-03-22

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

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

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 `base64url`

Base64URL parser and generator.

Availability:

`logtalk_load(base64(loader))`

Author: Paulo Moura

Version: 0:9:0

Date: 2021-03-10

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`

`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`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.11 base85

object

1.11.1 base85

Base85 encoder and decoder (Ascii85/btoa variant).

Availability:

`logtalk__load(base85(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2025-01-23

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

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

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.12 c45

object

1.12.1 c45

C4.5 decision tree learning algorithm. Builds a decision tree from a dataset object implementing the `dataset_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(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-20

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public classifier_protocol
```

Uses:

```
format
```

```
list
```

```
numberlist
```

```
pairs
```

Remarks:

- Algorithm: C4.5 is an extension of the ID3 algorithm that uses information gain ratio instead of information gain for attribute selection, which avoids bias towards attributes with many values.
- Discrete attributes: The learned decision tree is represented as `leaf(Class)` for leaf nodes and `tree(Attribute, Subtrees)` for internal nodes with discrete attributes, where `Subtrees` is a list of Value-Subtree pairs.
- Continuous attributes: For continuous (numeric) attributes, the tree uses binary threshold splits represented as `tree(Attribute, threshold(Threshold), LeftSubtree, RightSubtree)` where `LeftSubtree` corresponds to values \leq Threshold and `RightSubtree` to values $>$ Threshold.
- Missing values: Missing attribute values are represented using anonymous variables. During tree construction, examples with missing values for the split attribute are distributed to all branches. Entropy and gain calculations use only examples with known values for the attribute being evaluated.
- Tree pruning: The `prune/3` and `prune/5` predicates implement pessimistic error pruning (PEP), which estimates error rates using the upper confidence bound of the binomial distribution (Wilson score interval) with a configurable confidence factor (default 0.25, range (0.0, 1.0)) and minimum instances per leaf (default 2). Subtrees are replaced with leaf nodes when doing so would not increase the estimated error.

Inherited public predicates:

`classifier_to_clauses/4` `classifier_to_file/4` `learn/2` `predict/3` `print_classifier/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,+tree,+float,+positive_integer,-tree) - one`

Remarks:

- **Confidence factor:** The confidence factor controls the aggressiveness of pruning. It must be in the range (0.0, 1.0). Lower values result in more aggressive pruning (smaller, simpler trees), while higher values result in less pruning (larger, more complex trees). The default value is 0.25.
- **Minimum instances per leaf:** The minimum number of instances required at a leaf node. When a node has fewer instances than this value, the node may be pruned. It must be a positive integer. 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,+tree,-tree) - one`

Remarks:

- **Default parameters:** Uses the standard C4.5 default values: confidence factor of 0.25 (the confidence level for computing the upper bound of the error estimate) and minimum instances per leaf of 2.

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`, `knn`, `naive_bayes`, `nearest_centroid`, `random_forest`, `ada_boost`

1.13 cbor

object

1.13.1 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

1.13.2 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: 0:11:1

Date: 2021-12-06

Compilation flags:

```
static, context_switching_calls
```

Uses:

```
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`

`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
```

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.14 ccsds

object

1.14.1 ccsds

CCSDS Space Packet parser with no secondary header parsing. For secondary header support, use `ccsds(Length)` where `Length` is the secondary header size in bytes.

Availability:

```
logtalk_load(ccsds(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2025-12-05

Compilation flags:

```
static, context_switching_calls
```

Extends:

```
public ccsds(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/2 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

1.14.2 ccsds(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(loader))

Author: Paulo Moura

Version: 0:5:1

Date: 2026-02-04

Compilation flags:

static, context_switching_calls

Uses:

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/2
- 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/2`

Extracts time from a secondary header as `cuc_time(Coarse, Fine)` for CCSDS Unsegmented Time Code. Fails if no secondary header or time cannot be parsed.

Compilation flags:

`static`

Template:

`secondary_header_time(Packet,Time)`

Mode and number of proofs:

`secondary_header_time(+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

1.14.3 ccsds_types

Type definitions and arbitrary generators for CCSDS packets.

Availability:

`logtalk_load(ccsds(loader))`

Author: Paulo Moura

Version: 0:5:0

Date: 2025-12-15

Compilation flags:

`static`

Provides:

`type::type/1`

`type::check/2`

`arbitrary::arbitrary/1`

`arbitrary::arbitrary/2`

Uses:

`ccsds(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)

1.15 character_sets

category

1.15.1 character_set

Shared implementation support category for character set objects.

Availability:

```
logtalk_load(character_sets(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-05

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`
 - `word_bytes/4`
 - `dword_bytes/6`
 - `bytes_word/4`
 - `bytes_dword/6`
- 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

word_bytes/4

Converts a 16-bit word into two bytes using the given byte order.

Compilation flags:

static

Template:

word_bytes(Endian,Word,Byte1,Byte2)

Mode and number of proofs:

word_bytes(+atom,+integer,-integer,-integer) - one

dword_bytes/6

Converts a 32-bit word into four bytes using the given byte order.

Compilation flags:

static

Template:

dword_bytes(Endian,Word,Byte1,Byte2,Byte3,Byte4)

Mode and number of proofs:

dword_bytes(+atom,+integer,-integer,-integer,-integer,-integer) - one

`bytes_word/4`

Converts two bytes into a 16-bit word using the given byte order.

Compilation flags:

`static`

Template:

`bytes_word(Endian,Byte1,Byte2,Word)`

Mode and number of proofs:

`bytes_word(+atom,+integer,+integer,-integer) - one`

`bytes_dword/6`

Converts four bytes into a 32-bit word using the given byte order.

Compilation flags:

`static`

Template:

`bytes_dword(Endian,Byte1,Byte2,Byte3,Byte4,Word)`

Mode and number of proofs:

`bytes_dword(+atom,+integer,+integer,+integer,+integer,-integer) - one`

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.15.2 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 lower-casing 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

1.15.3 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

1.15.4 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

1.15.5 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

1.15.6 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

1.15.7 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

1.15.8 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

1.15.9 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

1.15.10 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

1.15.11 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

1.15.12 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

1.15.13 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

1.15.14 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

1.15.15 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

1.15.16 utf_16_character_set(Endian)

UTF-16 character set implementation parameterized by byte order.

Availability:

```
logtalk_load(character_sets(loader))
```

Author: Paulo Moura

Version: 1:0:0

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

1.15.17 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

1.15.18 `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

1.15.19 utf_32_character_set(Endian)

UTF-32 character set implementation parameterized by byte order.

Availability:

`logtalk_load(character_sets(loader))`

Author: Paulo Moura

Version: 1:0:0

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

1.15.20 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

1.15.21 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

1.15.22 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

1.15.23 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

1.15.24 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

1.15.25 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

1.15.26 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

1.15.27 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

1.15.28 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

1.15.29 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)

1.16 classifier_protocols

protocol

1.16.1 classifier_protocol

Protocol for machine learning classifiers.

Availability:

logtalk_load(classifier_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
 - learn/2
 - predict/3
 - classifier_to_clauses/4
 - classifier_to_file/4
 - print_classifier/1
- Protected predicates
- Private predicates
- Operators

Public predicates

learn/2

Learns a classifier from the given dataset object.

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`

`classifier_to_clauses/4`

Converts a classifier into a list of predicate clauses. Functor is the functor for the generated predicate clauses.

Compilation flags:

`static`

Template:

`classifier_to_clauses(Dataset,Classifier,Functor,Clauses)`

Mode and number of proofs:

`classifier_to_clauses(+object_identifier,+compound,+callable,-list(clause)) - one`

`classifier_to_file/4`

Exports a classifier to a file. Functor is the functor for the generated predicate clauses.

Compilation flags:

`static`

Template:

`classifier_to_file(Dataset,Classifier,Functor,File)`

Mode and number of proofs:

`classifier_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

`c45`, `isolation_forest`, `knn`, `naive_bayes`, `nearest_centroid`, `random_forest`

`protocol`

1.16.2 `dataset_protocol`

Protocol for datasets used with classifier algorithms.

Availability:

`logtalk_load(classifier_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)

1.17 code_metrics

object

1.17.1 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

1.17.2 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

1.17.3 `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

1.17.4 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

1.17.5 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

1.17.6 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

1.17.7 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

1.17.8 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

1.17.9 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

1.17.10 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

1.17.11 `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: $V = EL * \log_2(EV)$.
- 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

1.17.12 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

1.17.13 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: 1:0:0

Date: 2026-03-17

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

1.17.14 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

1.17.15 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

1.17.16 `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

1.17.17 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

1.17.18 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

1.17.19 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

1.17.20 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

1.17.21 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)

1.18 combinations

object

1.18.1 combinations

Implementation of combinations operations over lists.

Availability:

```
logtalk_load(combinations(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

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_with_replacement/3
combination_with_replacement/4 combinations/3 combinations/4
combinations_with_replacement/3 combinations_with_replacement/4 count_combinations/3
count_combinations_with_replacement/3 distinct_combination/3 distinct_combination/4
distinct_combinations/3 distinct_combinations/4 nth_combination/4 random_combination/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

1.18.2 combinations_protocol

Protocol for combinations operations over lists.

Availability:

logtalk_load(combinations(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
 - combinations/3
 - combination/3
 - combinations/4
 - combination/4
 - combinations_with_replacement/3
 - combinations_with_replacement/4
 - combination_with_replacement/3
 - combination_with_replacement/4
 - distinct_combinations/3
 - distinct_combination/3
 - distinct_combinations/4
 - distinct_combination/4
 - nth_combination/4
 - combination_index/4
 - count_combinations/3
 - count_combinations_with_replacement/3
 - random_combination/3
- Protected predicates
- Private predicates
- Operators

Public predicates

[combinations/3](#)

Generates all K-element combinations of a list.

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.

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`

`combinations_with_replacement/3`

Generates all K-element combinations with replacement.

Compilation flags:

static

Template:

`combinations_with_replacement(K,List,Combinations)`

Mode and number of proofs:

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

`combinations_with_replacement/4`

Generates all K-element combinations with replacement with the given order: default, lexicographic, or shortlex.

Compilation flags:

static

Template:

`combinations_with_replacement(K,List,Order,Combinations)`

Mode and number of proofs:

`combinations_with_replacement(+integer,+list,+atom,-list) - one`

`combination_with_replacement/3`

True iff the third argument is a K-element combination with replacement.

Compilation flags:

`static`

Template:

`combination_with_replacement(K,List,Combination)`

Mode and number of proofs:

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

`combination_with_replacement/4`

True iff the fourth argument is a K-element combination with replacement with the given order: default, lexicographic, or shortlex.

Compilation flags:

`static`

Template:

`combination_with_replacement(K,List,Order,Combination)`

Mode and number of proofs:

`combination_with_replacement(+integer,+list,+atom,-list) - one_or_more`

`distinct_combinations/3`

Generates all distinct K-element combinations of a list (deduplicating equal-valued combinations).

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.

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.

Compilation flags:

`static`

Template:

`nth_combination(K,List,Index,Combination)`

Mode and number of proofs:

`nth_combination(+integer,+list,+integer,-list) - zero_or_one`

`combination_index/4`

Returns the zero-based index of a combination.

Compilation flags:

`static`

Template:

`combination_index(K,List,Combination,Index)`

Mode and number of proofs:

`combination_index(+integer,+list,+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_combinations_with_replacement/3`

Counts the number of K-element combinations with replacement of a list.

Compilation flags:

`static`

Template:

`count_combinations_with_replacement(K,List,Count)`

Mode and number of proofs:

`count_combinations_with_replacement(+integer,+list,-integer) - 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

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.19 command_line_options

category

1.19.1 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: 1:0:0

Date: 2026-02-20

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

`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

1.19.2 `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: 1:0:0

Date: 2026-02-20

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 `ApplArguments` according to the option objects `OptionObjects` using default parsing options.

Compilation flags:

static

Template:

`parse(OptionObjects,ApplArguments,Options,PositionalArguments)`

Mode and number of proofs:

`parse(+list(object),+list(atom),-list,-list(atom)) - one_or_error`

parse/5

Parses the arguments `ApplArguments` 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,ApplArguments,Options,PositionalArguments,ParseOptions)`

Mode and number of proofs:

`parse(+list(object),+list(atom),-list,-list(atom),+list) - one_or_error`

[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`

[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), and `suppress_empty_meta(Boolean)` (default true).

Compilation flags:

`static`

Template:

`help(OptionObjects,Help,HelpOptions)`

Mode and number of proofs:

`help(+list(object),-atom,+list) - 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

`command_line_option`

1.20 core

category

1.20.1 `core_messages`

Logtalk core (compiler and runtime) default message tokenization.

Availability:

`built_in`

Author: Paulo Moura

Version: 1:146:0

Date: 2026-03-25

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

1.20.2 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

1.20.3 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

1.20.4 logtalk

Built-in object providing message printing, debugging, library, source file, and hacking methods.

Availability:

built_in

Author: Paulo Moura

Version: 3:4:0

Date: 2025-11-11

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
- 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`

`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

1.20.5 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

1.20.6 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)

1.21 coroutining

object

1.21.1 coroutining

Coroutining predicates.

Availability:

`logtalk__load(coroutining(loader))`

Author: Paulo Moura

Version: 0:5:0

Date: 2021-12-17

Compilation flags:

`static, context__switching__calls`

Dependencies:

(none)

Remarks:

- Supported backend Prolog systems: ECLiPSe, SICStus 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)

1.22 csv

object

1.22.1 csv

CSV files reading and writing predicates using the options Header - keep, Separator - comma, and Ignore-Quotes - 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

1.22.2 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

1.22.3 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

1.22.4 `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

1.22.5 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)

1.23 cuid2

object

1.23.1 cuid2

Cuid2 generator using atom representation, 24 symbols, and a lowercase alphanumeric alphabet.

Availability:

```
logtalk_load(cuid2(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

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`, `ulid`, `uuid`

object

1.23.2 `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: 1:0:0

Date: 2026-02-26

Compilation flags:

`static`, `context_switching_calls`

Implements:

public `cuid2_protocol`

Uses:

```
fast_random(Algorithm)
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

```
cuid2, ids(Representation,Bytes), ksuid(Representation,Alphabet), nanoid(Representation,Size,Alphabet),
snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds,TimestampBits,NodeBits,SequenceBits,Node),
ulid(Representation), uuid(Representation)
```

protocol

1.23.3 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)

1.24 datalog

object

1.24.1 datalog

Portable Datalog engine with stratified negation and incremental updates.

Availability:

logtalk_load(datalog(loader))

Author: Paulo Moura

Version: 0:1:1

Date: 2026-03-30

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`

1.24.2 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)

1.25 dates

object

1.25.1 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

1.25.2 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 `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 `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 \pm 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

1.25.3 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

1.25.4 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

1.26 dates_tz

object

1.26.1 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

1.26.2 `dates_tz_protocol`

Protocol for zone-aware date-time conversion using cached TZif data.

Availability:

`logtalk_load(dates_tz(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-08

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

`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`

`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
```

`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](#)

1.27 dead_code_scanner

object

1.27.1 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: 0:18:0

Date: 2026-04-05

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 known 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

1.27.2 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: 0:9:0

Date: 2026-03-19

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)

1.28 debug_messages

object

1.28.1 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)

1.29 debugger

object

1.29.1 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: 8:0:1

Date: 2026-02-07

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

1.29.2 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

1.29.3 debugger

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

1.29.4 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)

1.30 dependents

category

1.30.1 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

1.30.2 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](#)

1.31 dequeues

object

1.31.1 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

1.31.2 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)

1.32 diagrams

object

1.32.1 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

1.32.2 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

1.32.3 cytoscapejs_graph_language

Predicates for generating diagram files in the Cytoscape Exchange (CX2) JSON format.

Availability:

`logtalk_load(diagrams(loader))`

Author: Paulo Moura

Version: 1:1:0

Date: 2026-03-14

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

1.32.4 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: 1:5:0

Date: 2026-03-14

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

1.32.5 `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

1.32.6 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

1.32.7 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

1.32.8 `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

1.32.9 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`

1.32.10 `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

1.32.11 `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

1.32.12 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), brary_dependency_diagram(Format)	file_dependency_diagram(Format),	li-
---	----------------------------------	-----

object

1.32.13 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: 3:14:0
Date: 2026-03-14

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

1.32.14 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

1.32.15 `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

1.32.16 `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

1.32.17 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

1.32.18 `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

1.32.19 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

1.32.20 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), brary_dependency_diagram(Format)	directory_dependency_diagram(Format),	li-
--	---------------------------------------	-----

protocol

1.32.21 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

1.32.22 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

1.32.23 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

1.32.24 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

1.32.25 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

1.32.26 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`

1.32.27 `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

1.32.28 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

1.32.29 `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

<code>library_dependency_diagram(Format),</code>	<code>directory_dependency_diagram(Format),</code>
<code>file_dependency_diagram(Format),</code>	<code>entity_diagram(Format)</code>

object

1.32.30 mermaid_graph_language

Predicates for generating graph files using Mermaid.

Availability:

`logtalk_load(diagrams(loader))`

Author: Paulo Moura

Version: 0:5:0

Date: 2026-03-14

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

1.32.31 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

1.32.32 plantuml_graph_language

Predicates for generating diagrams in PlantUML format.

Availability:

`logtalk_load(diagrams(loader))`

Author: Paulo Moura

Version: 1:1:0

Date: 2026-03-14

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

1.32.33 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

1.32.34 `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

1.32.35 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

1.32.36 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),`
`uses_diagram(Format)`

`entity_diagram(Format),`

`inheritance_diagram(Format),`

1.33 dictionaries

object

1.33.1 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

1.33.2 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: 2:13:0

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
 - `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

1.33.3 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

1.33.4 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

1.33.5 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

1.33.6 `two3tree`

2-3 tree implementation.

Availability:

`logtalk_load(dictionaries(loader))`

Author: Michael T. Richter

Version: 1:0:0

Date: 2026-04-04

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

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`

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`

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`

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`

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`

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`

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`

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`

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`

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`

Mode and number of proofs:

`fix_node3_right_underflow(+term,+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`

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`

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`

Mode and number of proofs:

`insert_impl(+term,+term,+term,-term) - one`

`insert_empty/4`

Inserts into an empty tree.

Compilation flags:

`static`

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`

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`

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`

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`

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`

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`

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`

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`

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`

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`

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`

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`

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`

1.34 dif

object

1.34.1 dif

Provides `dif/2` and derived predicates.

Availability:

`logtalk_load(dif(loader))`

Author: Paulo Moura

Version: 1:3:0

Date: 2023-10-02

Compilation flags:

`static`, `context__switching__calls`

Dependencies:

(none)

Remarks:

- Supported backend Prolog systems: B-Prolog, ECLiPSe, SICStus 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)

1.35 doclet

object

1.35.1 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)

1.36 edcg

object

1.36.1 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`

1.37 events

object

1.37.1 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

1.37.2 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_registry`

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

1.37.3 `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

1.37.4 event_registry

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

1.37.5 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

1.37.6 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_registry`

1.38 expand_library_alias_paths

object

1.38.1 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)

1.39 expecteds

object

1.39.1 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

1.39.2 `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

1.39.3 `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`

1.40 fcube

object

1.40.1 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

1.41 flags

category

1.41.1 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

Compilation flags:

static

validate_type/1

Compilation flags:

static

is_validator/1

Compilation flags:

static

Operators

(none)

protocol

1.41.2 flags_validator

Flag validation protocol. Must be implemented by validator objects.

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
 - 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)

1.42 format

object

1.42.1 format

Formatted output predicates.

Availability:

logtalk_load(format(loader))

Author: Paulo Moura

Version: 1:3:0

Date: 2025-11-18

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)

1.43 genint

object

1.43.1 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

1.43.2 `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)

1.44 gensym

object

1.44.1 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

1.44.2 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)

1.45 geospatial

object

1.45.1 geospatial

Geospatial predicates over geographic coordinates represented as (Latitude,Longitude).

Availability:

logtalk_load(geospatial(loader))

Author: Paulo Moura

Version: 0:2:0

Date: 2026-02-25

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_expand/3 bbox_from_coordinates/2
 bbox_intersects/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 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
 - distance/4
 - distance/5
- Protected predicates
- Private predicates
- Operators

Public predicates

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

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

1.45.2 geospatial_protocol

Geospatial predicates protocol.

Availability:

logtalk_load(geospatial(loader))

Author: Paulo Moura

Version: 0:2:0

Date: 2026-02-25

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_union/3
- bbox_expand/3
- bbox_from_coordinates/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 (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 is inside or on the boundary of a bounding box term `bbox((MinLatitude,MinLongitude),(MaxLatitude,MaxLongitude))`.

Compilation flags:

static

Template:

bbox_contains(BoundingBox,Coordinate)

Mode and number of proofs:

bbox_contains(+compound,+compound) - zero_or_one

`bbox_intersects/2`

True when two bounding boxes intersect or touch.

Compilation flags:

`static`

Template:

`bbox_intersects(BoundingBox1,BoundingBox2)`

Mode and number of proofs:

`bbox_intersects(+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_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`

`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((MinLatitude,MinLongitude),(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](#), [numberlist](#), [listp](#)

1.46 git

object

1.46.1 git

Predicates for accessing a git project current branch and latest commit data.

Availability:

`logtalk_load(git(loader))`

Author: Paulo Moura

Version: 2:1:2

Date: 2024-03-11

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

- Public predicates
- Protected predicates
- Private predicates
- Operators

Public predicates

(no local declarations; see entity ancestors 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

1.46.2 git_protocol

Predicates for accessing a git project current branch and latest commit data.

Availability:

logtalk_load(git(loader))

Author: Paulo Moura

Version: 1:1:0

Date: 2022-01-21

Compilation flags:

static

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- Public predicates
 - branch/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

`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)

1.47 grammars

object

1.47.1 `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

1.47.2 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

1.47.3 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 or 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

1.47.4 `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)

1.48 graphs

category

1.48.1 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

1.48.2 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

1.48.3 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: 1:0:0

Date: 2026-02-25

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

1.48.4 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 $|V| * \log_2(|V|)$. The cutoff $|E| = |V| * \log_2(|V|)$ 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

1.48.5 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

1.48.6 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: 1:0:0

Date: 2026-02-25

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_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_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

1.48.7 `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

1.48.8 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: 1:0:0

Date: 2026-02-20

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

1.48.9 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

1.48.10 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

1.48.11 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

1.48.12 `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: 1:0:0

Date: 2026-02-20

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_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
 - is_connected/1
 - 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

is_connected/1

True iff the graph is connected (every vertex is reachable from every other vertex).

Compilation flags:

static

Template:

`is_connected(Graph)`

Mode and number of proofs:

`is_connected(+graph) - 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

1.48.13 `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

1.48.14 `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: 1:0:0

Date: 2026-02-20

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

1.48.15 `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: 1:0:0

Date: 2026-02-25

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

1.48.16 `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

1.48.17 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

1.48.18 `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: 1:0:0

Date: 2026-02-20

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_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
 - is_connected/1
 - 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

`is_connected/1`

True if 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

`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)

1.49 hashes

object

1.49.1 `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: 1:0:0

Date: 2026-04-05

Compilation flags:

`static, context_switching_calls`

Implements:

`public hash_protocol`

Uses:

`hash_common_32`

`list`

Remarks:

`(none)`

Inherited public predicates:

`hash/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

`crc32_reflected(Polynomial)`, `crc32b`, `crc32c`, `crc32posix`, `crc32mpeg2`, `crc32bzip2`, `crc32q`, `mur-murhash3_x86_32`, `fnv1a_32`

object

1.49.2 `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: 1:0:0

Date: 2026-04-05

Compilation flags:

`static`, `context_switching_calls`

Implements:

public `hash_protocol`

Uses:

`hash_common_32`

Remarks:

(none)

Inherited public predicates:

hash/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

`crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength)`, `crc32b`, `crc32c`, `crc32posix`,
`crc32mpeg2`, `crc32bzip2`, `crc32q`, `murmurhash3_x86_32`, `fnv1a_32`

object

1.49.3 `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:

hash/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

crc32_reflected(Polynomial), crc32c, crc32posix, crc32mpeg2, crc32bzip2, crc32q, mur-
murhash3_x86_32, fnv1a_32

object

1.49.4 `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:

```
hash/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

`crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength)`, `crc32mpeg2`, `crc32posix`, `crc32q`,
`crc32_reflected(Polynomial)`, `crc32b`, `crc32c`, `murmurhash3_x86_32`, `fnv1a_32`

object

1.49.5 `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:

[hash/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

`crc32_reflected(Polynomial)`, `crc32b`, `crc32posix`, `crc32mpeg2`, `crc32bzip2`, `crc32q`, `mur-murhash3_x86_32`, `fnv1a_32`

object

1.49.6 `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:

hash/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

crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength), crc32posix, crc32bzip2, crc32q,
crc32_reflected(Polynomial), crc32b, crc32c, murmurhash3_x86_32, fnv1a_32

object

1.49.7 `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:

```
hash/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

`crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength)`, `crc32_reflected(Polynomial)`, `crc32b`, `crc32c`, `crc32mpeg2`, `crc32bzip2`, `crc32q`, `murmurhash3_x86_32`, `fnv1a_32`

object

1.49.8 `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:

[hash/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

`crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength)`, `crc32mpeg2`, `crc32bzip2`, `crc32posix`, `crc32_reflected(Polynomial)`, `crc32b`, `crc32c`, `murmurhash3_x86_32`, `fnv1a_32`

object

1.49.9 djb2_32

DJB2 32-bit hash function.

Availability:

`logtalk_load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static`, `context_switching_calls`

Implements:

`public hash_protocol`

Uses:

`hash_common_32`

Remarks:

(none)

Inherited public predicates:

`hash/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

`djb2_64`, `sdbm_32`, `fnv1a_32`

object

1.49.10 djb2_64

DJB2 64-bit hash function.

Availability:

`logtalk_load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static, context_switching_calls`

Implements:

`public hash_protocol`

Uses:

`hash_common_64`

Remarks:

(none)

Inherited public predicates:

`hash/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

`djb2_32`, `sdbm_64`, `fnv1a_64`

object

1.49.11 `fips202_hash(A,B,C)`

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: 1:0:0

Date: 2026-04-04

Compilation flags:

`static`, `context_switching_calls`

Implements:

`public hash_protocol`

Uses:

`hash_common_32`

`hash_common_64`

`list`

Remarks:

(none)

Inherited public predicates:

`hash/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

1.49.12 `fnv1a_32`

FNV-1a 32-bit hash function.

Availability:

`logtalk_load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static, context_switching_calls`

Implements:

public hash_protocol

Uses:

hash_common_32

Remarks:

(none)

Inherited public predicates:

hash/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

fnv1a_64, djb2_32, sdbm_32

object

1.49.13 `fnv1a_64`

FNV-1a 64-bit hash function.

Availability:

`logtalk_load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static, context_switching_calls`

Implements:

`public hash_protocol`

Uses:

`hash_common_64`

Remarks:

(none)

Inherited public predicates:

`hash/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

[fnv1a_32](#), [djb2_64](#), [sdbm_64](#)

object

1.49.14 `hash_common_32`

Auxiliary predicates for the hashes library 32-bit algorithms.

Availability:

`logtalk_load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static, context_switching_calls`

Uses:

[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`
 - `little_endian_word32/2`
 - `big_endian_word32/2`
 - `integer_to_little_endian_bytes32/2`
 - `integer_to_big_endian_bytes32/2`
 - `pad_md/4`
- 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} .

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

little_endian_word32/2

Decodes four bytes in little-endian order into a 32-bit word.

Compilation flags:

static

Template:

little_endian_word32(Bytes,Word)

Mode and number of proofs:

little_endian_word32(+list(integer),-integer) - one

`big_endian_word32/2`

Decodes four bytes in big-endian order into a 32-bit word.

Compilation flags:

`static`

Template:

`big_endian_word32(Bytes,Word)`

Mode and number of proofs:

`big_endian_word32(+list(integer),-integer) - one`

`integer_to_little_endian_bytes32/2`

Encodes a 32-bit word into four bytes in little-endian order.

Compilation flags:

`static`

Template:

`integer_to_little_endian_bytes32(Integer,Bytes)`

Mode and number of proofs:

`integer_to_little_endian_bytes32(+integer,-list(integer)) - one`

`integer_to_big_endian_bytes32/2`

Encodes a 32-bit word into four bytes in big-endian order.

Compilation flags:

`static`

Template:

`integer_to_big_endian_bytes32(Integer,Bytes)`

Mode and number of proofs:

`integer_to_big_endian_bytes32(+integer,-list(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`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.49.15 `hash_common_64`

Auxiliary predicates for the hashes library 64-bit algorithms.

Availability:

`logtalk_load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static, context_switching_calls`

Dependencies:

(none)

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
 - integer_to_big_endian_bytes64/2
- 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

integer_to_big_endian_bytes64/2

Encodes a 64-bit word into eight bytes in big-endian order.

Compilation flags:

static

Template:

integer_to_big_endian_bytes64(Integer,Bytes)

Mode and number of proofs:

integer_to_big_endian_bytes64(+integer,-list(integer)) - one

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

protocol

1.49.16 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: 1:0:0

Date: 2026-04-04

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

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

object

1.49.17 md5

MD5 hash function.

Availability:

logtalk_load(hashes(loader))

Author: Paulo Moura

Version: 1:0:0
Date: 2026-04-04

Compilation flags:
static, context_switching_calls

Implements:
public hash_protocol

Uses:
hash_common_32
list

Remarks:
(none)

Inherited public predicates:
hash/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

[sha1](#), [sha256](#)

object

1.49.18 [murmurhash3_x64_128](#)

MurmurHash3 x64 128-bit hash function with seed 0.

Availability:

`logtalk_load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static, context_switching_calls`

Implements:

`public hash_protocol`

Uses:

[hash_common_64](#)

[list](#)

Remarks:

(none)

Inherited public predicates:

[hash/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

`murmurhash3_x86_32`, `murmurhash3_x86_128`

object

1.49.19 `murmurhash3_x86_128`

MurmurHash3 x86 128-bit hash function with seed 0.

Availability:

`logtalk__load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static`, `context_switching_calls`

Implements:

`public hash_protocol`

Uses:

`hash_common_32`

list

Remarks:

(none)

Inherited public predicates:

hash/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

murmurhash3_x86_32, murmurhash3_x64_128

object

1.49.20 murmurhash3_x86_32

MurmurHash3 x86 32-bit hash function with seed 0.

Availability:

`logtalk_load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static, context_switching_calls`

Implements:

`public hash_protocol`

Uses:

`hash_common_32`

`list`

Remarks:

`(none)`

Inherited public predicates:

`hash/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

[murmurhash3_x86_128](#), [murmurhash3_x64_128](#)

object

1.49.21 `sdbm_32`

sdbm 32-bit hash function.

Availability:

`logtalk__load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static, context_switching_calls`

Implements:

public [hash_protocol](#)

Uses:

[hash_common_32](#)

Remarks:

(none)

Inherited public predicates:

`hash/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

`sdbm_64`, `djb2_32`, `fnv1a_32`

object

1.49.22 `sdbm_64`

`sdbm` 64-bit hash function.

Availability:

`logtalk_load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

static, context_switching_calls

Implements:

public hash_protocol

Uses:

hash_common_64

Remarks:

(none)

Inherited public predicates:

hash/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

sdbm_32, djb2_64, fnv1a_64

object

1.49.23 sha1

SHA-1 hash function.

Availability:

```
logtalk_load(hashes(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public hash_protocol
```

Uses:

```
hash_common_32
```

```
list
```

Remarks:

(none)

Inherited public predicates:

```
hash/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

[md5](#), [sha256](#)

object

1.49.24 [sha256](#)

SHA-256 hash function.

Availability:

`logtalk__load(hashes(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-04

Compilation flags:

`static, context_switching_calls`

Implements:

`public` [hash_protocol](#)

Uses:

[hash_common_32](#)

[list](#)

Remarks:

(none)

Inherited public predicates:

[hash/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

[md5](#), [sha1](#)

object

1.49.25 [sha3__224](#)

FIPS 202 SHA3-224 hash 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(144,6,28)

Remarks:
(none)

Inherited public predicates:
hash/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

`sha3_256`, `sha3_384`, `sha3_512`, `shake128(OutputBytes)`, `shake256(OutputBytes)`

object

1.49.26 `sha3_256`

FIPS 202 SHA3-256 hash 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(136,6,32)`

Remarks:

(none)

Inherited public predicates:

`hash/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

sha3_224, sha3_384, sha3_512, sha256, shake128(OutputBytes), shake256(OutputBytes)

object

1.49.27 sha3_384

FIPS 202 SHA3-384 hash 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(104,6,48)`

Remarks:

(none)

Inherited public predicates:

hash/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

sha3_224, sha3_256, sha3_512, shake128(OutputBytes), shake256(OutputBytes)

object

1.49.28 sha3_512

FIPS 202 SHA3-512 hash 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(72,6,64)

Remarks:

(none)

Inherited public predicates:

hash/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

sha3_224, sha3_256, sha3_384, shake128(OutputBytes), shake256(OutputBytes)

object

1.49.29 `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:

```
hash/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

`shake256(OutputBytes)`, `sha3_224`, `sha3_256`, `sha3_384`, `sha3_512`

object

1.49.30 `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:

`hash/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

shake128(OutputBytes), sha3_224, sha3_256, sha3_384, sha3_512

object

1.49.31 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:

`hash/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

`siphash_2_4(Key)`, `fnv1a_64`, `crc32b`, `crc32c`

object

1.49.32 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: 1:0:0

Date: 2026-04-04

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public hash_protocol
```

Uses:

```
hash_common_64
```

```
list
```

Remarks:

```
(none)
```

Inherited public predicates:

```
hash/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

siphash_2_4, fnv1a_64, crc32b, crc32c

1.50 heaps

object

1.50.1 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

1.50.2 `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

1.50.3 `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

1.50.4 `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

1.50.5 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

1.50.6 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

1.50.7 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

1.50.8 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

1.50.9 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: 1:0:0

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

`pairing_heap_min`, `pairing_heap_max`

object

1.50.10 `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

1.50.11 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)`

1.51 help

object

1.51.1 help

Command-line help for Logtalk tools, libraries, entities, predicates, and non-terminals.

Availability:

`logtalk_load(help(loader))`

Author: Paulo Moura

Version: 0:43:0

Date: 2026-02-04

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)

1.52 hierarchies

`category`

1.52.1 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_hierarchy`

Remarks:

(none)

Inherited public predicates:

`ancestor/1 ancestor/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

1.52.2 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

1.52.3 hierarchy

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

1.52.4 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

1.52.5 proto_hierarchy

Prototype hierarchy protocol.

Availability:

`logtalk_load(hierarchies(loader))`

Author: Paulo Moura

Version: 1:1:0

Date: 2006-02-20

Compilation flags:

`static`

Extends:

`public hierarchy`

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`

1.53 hook_flows

object

1.53.1 `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

1.53.2 `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)`

1.54 hook_objects

object

1.54.1 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

1.54.2 `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

1.54.3 `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

1.54.4 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`, `pro-`
`log_module_hook(Module)`, `object_wrapper_hook`, `write_to_stream_hook(Stream,Options)`,
`write_to_stream_hook(Stream)`, `print_goal_hook`, `suppress_goal_hook`

object

1.54.5 `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)`, `back-end_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

1.54.6 `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, sup-
press_goal_hook
```

object

1.54.7 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

1.54.8 `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

1.54.9 `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

1.54.10 `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

1.54.11 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

1.54.12 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, sup-
press_goal_hook

object

1.54.13 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

1.54.14 `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`

1.55 html

category

1.55.1 html

HTML generation.

Availability:

`logtalk_load(html(loader))`

Author: Paul Brown and Paulo Moura

Version: 0:3:0

Date: 2021-03-30

Compilation flags:

`static`

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- Public predicates
 - generate/2
 - void_element/1
 - normal_element/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

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`

Protected predicates

(none)

Private predicates

`doctype/1`

Doctype text.

Compilation flags:

`static`

Template:

`doctype(DocType)`

Mode and number of proofs:

`doctype(?atom) - one`

Operators

(none)

object

1.55.2 `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:

`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

1.55.3 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:

`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)

1.56 ids

object

1.56.1 ids

Generator of random identifiers represented as atoms with 160 bits (20 bytes) of randomness.

Availability:

```
logtalk_load(ids(loader))
```

Author: Paulo Moura

Version: 1:2:0

Date: 2026-02-26

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, uuid, ulid

object

1.56.2 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: 1:1:0

Date: 2026-02-26

Compilation flags:

```
static, context_switching_calls
```

Uses:

```
base64
fast_random
list
os
```

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),`
`ulid(Representation)`

1.57 intervals

object

1.57.1 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: 1:2:1

Date: 2022-01-15

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public intervalp
```

Aliases:

```
intervalp before/2 as b/2
intervalp after/2 as bi/2
intervalp meets/2 as m/2
intervalp met_by/2 as mi/2
intervalp overlaps/2 as o/2
intervalp overlapped_by/2 as oi/2
intervalp starts/2 as s/2
intervalp started_by/2 as si/2
intervalp during/2 as d/2
intervalp contains/2 as di/2
intervalp finishes/2 as f/2
intervalp finished_by/2 as fi/2
intervalp 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 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)

protocol

1.57.2 intervalp

Basic temporal interval relations protocol (based on James F. Allen Interval Algebra work).

Availability:

`logtalk_load(intervals(loader))`

Author: Paulo Moura

Version: 1:1:0

Date: 2014-04-26

Compilation flags:

`static`

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- Public predicates
 - new/3
 - valid/1
 - 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

[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](#)

1.58 iso8601

object

1.58.1 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: 1:3:0

Date: 2026-04-08

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('Thh:mm:ss',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('YYYYMMDDThh:mm:ssZ',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 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'
```

`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'`

`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)

1.59 isolation_forest

object

1.59.1 isolation_forest

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 a dataset object implementing the dataset_protocol protocol. Missing attribute values are represented using anonymous variables.

Availability:

```
logtalk_load(isolation_forest(loader))
```

Author: Paulo Moura

Version: 1:0:1

Date: 2026-03-11

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public classifier_protocol
```

Imports:

```
public options
```

Uses:

```
fast_random(Algorithm)
format
integer
list
numberlist
pairs
type
```

Remarks:

- Algorithm: The Extended Isolation Forest builds an ensemble of isolation trees (iTrees) by recursively partitioning the data using random hyperplanes. Anomalous points, being few and different, require fewer partitions (shorter path lengths) to be isolated.
- Extended vs Original: The original Isolation Forest uses axis-aligned splits (random attribute + random value), which introduces bias in anomaly scores along coordinate axes. The extended version uses random hyperplane cuts with arbitrary slopes, producing more consistent and reliable anomaly scores.

- Extension level: The extension level controls the dimensionality of the random hyperplane cuts. Level 0 corresponds to the original axis-aligned Isolation Forest. The default level is $d - 1$ (fully extended) where d is the number of dimensions.
- Prediction: The `predict/3` predicate returns anomaly if the anomaly score is above the threshold (default: 0.5) and normal otherwise. The `score_all/3` predicate returns a sorted list of all instances with their corresponding scores and class labels. Predictions use by default the learned model options but can override them using the `anomaly_threshold/1` option.
- Anomaly score: The anomaly score $s(x)$ is computed as $s(x) = 2^{(-E(h(x))/c(\psi))}$ where $E(h(x))$ is the average path length across all trees, $c(\psi)$ is the average path length of unsuccessful searches in a BST, and ψ is the subsample size. Scores close to 1 indicate anomalies; scores below 0.5 indicate normal points.
- Discrete attributes: Discrete (categorical) attributes are mapped to numeric indices based on their position in the attribute value list declared by the dataset. This allows the algorithm to handle datasets with mixed attribute types.
- Missing values: Missing attribute values are represented using anonymous variables. During tree construction, missing values are replaced with random values drawn from the observed range of the corresponding attribute. During scoring, instances with missing values are sent down both branches of the tree and the path length is computed as the weighted average of the two branches.
- Classifier representation: The learned model is represented as an `if_model(Trees, SubsampleSize, AttributeNames, Attributes, Ranges, Options)` compound term.

Inherited public predicates:

```
check_option/1 check_options/1 classifier_to_clauses/4 classifier_to_file/4 default_option/1
default_options/1 learn/2 option/2 option/3 predict/3 print_classifier/1 valid_option/1
valid_options/1
```

- Public predicates
 - `learn/3`
 - `predict/4`
 - `score/3`
 - `score_all/3`
- Protected predicates
- Private predicates
- Operators

Public predicates

learn/3

Learns an isolation forest model from the given dataset object using the specified options. Valid options are `number_of_trees/1` (default: 100), `subsample_size/1` (default: 256 or the number of instances if smaller), `extension_level/1` (default: $d - 1$ where d is the number of dimensions), and `anomaly_threshold/1` (default: 0.5).

Compilation flags:

`static`

Template:

`learn(Dataset,Model,Options)`

Mode and number of proofs:

`learn(+object_identifier,-compound,+list(compound)) - one`

predict/4

Predicts whether an instance is an anomaly or normal using the learned model and the anomaly threshold with the given options. The instance is a list of Attribute-Value pairs where missing values are represented using anonymous variables. Returns `anomaly` if the anomaly score is above the threshold, `normal` otherwise.

Compilation flags:

`static`

Template:

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

Mode and number of proofs:

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

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

[dataset_protocol](#), [c45](#), [random_forest](#), [ada_boost](#)

1.60 issue_creator

object

1.60.1 issue_creator

Support for automatically creating bug report issues for failed tests in GitHub or GitLab servers.

Availability:

```
logtalk_load(issue_creator(loader))
```

Author: Paulo Moura

Version: 0:12:1

Date: 2025-03-03

Compilation flags:

```
static, context_switching_calls
```

Provides:

```
logtalk::message_hook/4
```

Uses:

```
git  
os  
term_io  
user
```

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
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.61 java

object

1.61.1 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

1.61.2 `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

1.61.3 `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

1.61.4 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

1.61.5 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

1.61.6 `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)

1.62 json

object

1.62.1 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

1.62.2 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

1.62.3 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: 0:14:0

Date: 2026-01-25

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

1.62.4 json_protocol

JSON parser and generator protocol.

Availability:

```
logtalk_load(json(loader))
```

Author: Paulo Moura and Jacinto Dávila

Version: 0:11:0

Date: 2022-11-09

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 the JSON contents read from the given source (`codes(List)`, `stream(Stream)`, `line(Stream)`, `file(Path)`, `chars(List)`, or `atom(Atom)`) into a term. Fails if the JSON contents cannot be parsed.

Compilation flags:

```
static
```

Template:

```
parse(Source,Term)
```

Mode and number of proofs:

```
parse(++compound,--term) - one_or_error
```

`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
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.63 json_ld

object

1.63.1 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 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

1.63.2 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 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

1.63.3 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: 1:0:0

Date: 2026-02-05

Compilation flags:

static, context_switching_calls

Implements:

public json_ld_protocol

Uses:

json(ObjectRepresentation,PairRepresentation,StringRepresentation)
list

Remarks:

(none)

Inherited public predicates:

compact/3 expand/2 flatten/2 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

1.63.4 json_ld_protocol

JSON-LD 1.1 parser, generator, and processor protocol.

Availability:

```
logtalk_load(json_ld(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
 - parse/2
 - generate/2
 - expand/2
 - compact/3
 - flatten/2
- 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`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.64 json_lines

object

1.64.1 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

1.64.2 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

1.64.3 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: 1:1:0

Date: 2026-01-25

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

1.64.4 json_lines_protocol

JSON Lines parser and generator protocol.

Availability:

`logtalk_load(json_lines(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2025-05-27

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)

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)

1.65 json_rpc

object

1.65.1 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: 1:0:0

Date: 2026-02-24

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

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 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 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)

1.66 json_schema

object

1.66.1 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

1.66.2 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

1.66.3 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: 1:1:0

Date: 2026-03-23

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public json_schema_protocol
```

Uses:

```
json(ObjectRepresentation,PairRepresentation,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

1.66.4 json_schema_protocol

JSON Schema parser and validator protocol.

Availability:

```
logtalk_load(json_schema(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-01-29

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`

`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)`

1.67 knn

object

1.67.1 knn

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(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-20

Compilation flags:

`static, context_switching_calls`

Implements:

`public classifier_protocol`

Imports:

`public options`

Uses:

`format`

list
pairs
type

Remarks:

- Algorithm: k-NN is a lazy learning algorithm that classifies instances based on the majority class among the k nearest training instances.
- Distance metrics: Supports Euclidean, Manhattan, Chebyshev, and Minkowski distance metrics.
- Weighting schemes: Supports uniform, distance-based, and Gaussian weighting of neighbors.
- Feature types: Automatically handles numeric and categorical features.
- Classifier representation: The learned classifier is represented (by default) as a `knn_classifier(AttributeNames, FeatureTypes, Instances)` where `Instances` contains the training data.

Inherited public predicates:

`check_option/1` `check_options/1` `classifier_to_clauses/4` `classifier_to_file/4` `default_option/1`
`default_options/1` `learn/2` `option/2` `option/3` `predict/3` `print_classifier/1` `valid_option/1`
`valid_options/1`

- Public predicates
 - `predict/4`
 - `predict_probabilities/3`
 - `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/3](#)

Predicts class probabilities for a new instance using the learned classifier and default 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)
```

Mode and number of proofs:

```
predict_probabilities(+compound,+list,-list) - 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`, `isolation_forest`, `naive_bayes`, `nearest_centroid`, `random_forest`, `ada_boost`

1.68 ksuid

object

1.68.1 ksuid

KSUID generator using atom representation and the canonical Base62 alphabet.

Availability:

```
logtalk__load(ksuid(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

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`, `nanoid`, `ids`, `ulid`, `snowflakeid`, `uuid`

object

1.68.2 `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: 1:0:0

Date: 2026-02-26

Compilation flags:

static, context__switching__calls

Implements:

public ksuid__protocol

Uses:

fast_random(Algorithm)

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), nanoid(Representation,Size,Alphabet),
ids(Representation,Bytes), ulid(Representation), snowflakeid(Representation,EpochMilliseconds,TimeUnit,Milliseconds,TimeUnit),
uuid(Representation)

protocol

1.68.3 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)

1.69 lgtdoc

object

1.69.1 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: 11:3:0

Date: 2026-03-31

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

1.69.2 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

1.69.3 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.xml')])
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.xml')])
```

```
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.xml')])
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.xml')])
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

1.70 lgtunit

object

1.70.1 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

1.70.2 `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: 1:0:0

Date: 2026-03-13

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_/7`
 - `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_/7

Table of predicate coverage data.

Compilation flags:

dynamic

Template:

method_(Entity,Predicate,Line,Covered,Total,Percentage,Clauses)

Mode and number of proofs:

method_(?entity_identifier,?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

1.70.3 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: 3:2:1

Date: 2026-02-06

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

1.70.4 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

1.70.5 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

1.70.6 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: 1:0:0
Date: 2026-03-13

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_/7`
- 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_/7`

Table of predicate coverage data.

Compilation flags:
dynamic

Template:

```
predicate_coverage_(Entity,Predicate,Line,Covered>Total,Percentage,Clauses)
```

Mode and number of proofs:

```
predicate_coverage_(?entity_identifier,?predicate_indicator,?integer,?integer,?integer,?float,?list) -  
zero_or_more
```

Operators

(none)

object

1.70.7 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: 22:7:0

Date: 2026-04-01

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
```

```
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
 - 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 $\text{abs}(\text{Number1} - \text{Number2}) \leq \max(\text{abs}(\text{Number1}), \text{abs}(\text{Number2})) * \text{epsilon}$. 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 $\text{abs}(\text{Number1} - \text{Number2}) \leq \max(\text{abs}(\text{Number1}), \text{abs}(\text{Number2})) * \text{Epsilon}$. 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 $\text{abs}(\text{Number1} - \text{Number2}) \leq \min(\text{abs}(\text{Number1}), \text{abs}(\text{Number2})) * \text{Epsilon}$. 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 $\text{abs}(\text{Number1} - \text{Number2}) \leq \max(\text{RelativeTolerance} * \max(\text{abs}(\text{Number1}), \text{abs}(\text{Number2})), \text{AbsoluteTolerance})$. 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`

`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

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

1.70.8 lgtunit__messages

Logtalk unit test framework default message translations.

Availability:

logtalk_load(lgtunit(loader))

Author: Paulo Moura

Version: 12:2:0

Date: 2025-10-20

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

1.70.9 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

1.70.10 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

1.70.11 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

1.70.12 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

1.70.13 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

1.70.14 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

1.70.15 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

1.70.16 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

1.70.17 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

1.70.18 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

1.70.19 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)

1.71 library

protocol

1.71.1 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

1.71.2 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

1.71.3 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
 - (\leq)/2
 - (\geq)/2
- Protected predicates
- Private predicates
- Operators
 - op(100,xfx, \leq)
 - op(100,xfx, \geq)

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

1.72 linda

object

1.72.1 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

1.72.2 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: 2:0:0

Date: 2026-03-27

Compilation flags:

`static`

Extends:

`public options`

Uses:

`os`

`socket`

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. Call `close_client/1` after this predicate.

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

1.72.3 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: 2:0:0

Date: 2026-03-27

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
 - 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`

`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)`

1.73 linter_reporter

object

1.73.1 linter_reporter

Intercepts compiler linter warnings and caches them as machine-readable diagnostics.

Availability:

`logtalk_load(linter_reporter(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-03-30

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`

`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)

1.74 listing

category

1.74.1 listing

Listing predicates.

Availability:

logtalk_load(listing(loader))

Author: Paulo Moura

Version: 1:0:0

Date: 2024-01-26

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 ground:

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)

1.75 logging

object

1.75.1 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

1.75.2 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

1.75.3 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

1.76 loops

object

1.76.1 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

1.76.2 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

1.77 mcp_server

protocol

1.77.1 mcp_prompt_protocol

Protocol for Logtalk objects that provide prompts to be exposed via an MCP (Model Context Protocol) server. Implements the MCP 2025-06-18 specification. 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.

Availability:

```
logtalk_load(mcp_server(loader))
```

Author: Paulo Moura

Version: 0:2:0

Date: 2026-02-24

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.

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

1.77.2 mcp_resource_protocol

Protocol for Logtalk objects that provide resources to be exposed via an MCP (Model Context Protocol) server. Implements the MCP 2025-06-18 specification. Implementing objects must define the set of resources available and handle resource read requests. Resources expose data and content from the application that MCP clients can access.

Availability:

```
logtalk_load(mcp_server(loader))
```

Author: Paulo Moura

Version: 0:2:0

Date: 2026-02-24

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` 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 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.

Inherited public predicates:

(none)

- Public predicates
 - `resources/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_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

1.77.3 mcp_server

MCP (Model Context Protocol) server for Logtalk applications. Exposes Logtalk objects implementing the mcp_tool_protocol and optionally mcp_prompt_protocol and mcp_resource_protocol protocols as MCP tool, prompt, and resource providers over stdio transport with Content-Length framing. Implements the MCP 2025-03-26 specification. Uses the json_rpc library for JSON-RPC 2.0 message handling.

Availability:

logtalk_load(mcp_server(loader))

Author: Paulo Moura

Version: 0:5:1

Date: 2026-03-13

Compilation flags:

static, context_switching_calls

Imports:

public options

Uses:

json_rpc

list

term_io

user

Remarks:

- MCP specification: Implements the Model Context Protocol 2025-03-26: <https://spec.modelcontextprotocol.io/specification/2025-03-26/>
- Transport: Uses stdio (standard input/output) with Content-Length header framing as defined by the MCP specification.
- Version negotiation: During initialization, the server performs protocol version negotiation. The client sends its highest supported version in the initialize request. The server selects the highest version it supports that is compatible (i.e., less than or equal to the client version). If no compatible version exists, the server responds with error -32602 (“Unsupported protocol version”) including a data field with the list of supported versions.
- Capabilities: Supports the tools capability (tools/list and tools/call). Optionally supports the prompts capability (prompts/list and prompts/get) when the application declares it via capabilities/1 and implements mcp_prompt_protocol. Optionally supports the resources capability (resources/list and resources/read) when the application declares it via capabilities/1 and implements mcp_resource_protocol. Optionally supports the elicitation capability (elicitation/create) when the application declares it via capabilities/1.
- Tool discovery: Tools are discovered from objects implementing the mcp_tool_protocol. Tool metadata (descriptions, parameter schemas) is derived from info/2 and mode/2 directives.
- Prompt discovery: Prompts are discovered from objects implementing the mcp_prompt_protocol. Prompt metadata (names, descriptions, arguments) is declared via the prompts/1 predicate.
- Resource discovery: Resources are discovered from objects implementing the mcp_resource_protocol. Resource metadata (URIs, names, descriptions, MIME types) is declared via the resources/1 predicate.
- Auto-dispatch: When an object does not define tool_call/3 or tool_call/4 for a tool, the server auto-dispatches: it calls the predicate as a message to the object, collects output-mode arguments, and returns them as text content.
- Elicitation: When the application declares elicitation in its capabilities/1, the server constructs an elicit closure and passes it to tool_call/4. The closure sends elicitation/create requests to the client and reads back the user response. This enables interactive tools that can ask the user questions during execution.
- Error handling: Predicate failures and exceptions both result in MCP tool-level errors with isError set to true.

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
 - elicit_request/5
- Protected predicates
- Private predicates
 - initialized_/0
 - application_/1
 - server_name_/1
 - server_version_/1
 - server_title_/1
 - elicit_counter_/1
 - application_capabilities_/1
- Operators

Public predicates

start/2

Starts the MCP server with the given server name and application object. The application object must implement the `mcp_tool_protocol` protocol. Reads JSON-RPC messages from standard input and writes responses to standard output using Content-Length framing. Blocks until the client disconnects or an exit signal is received.

Compilation flags:

```
static
```

Template:

```
start(Name,Application)
```

Mode and number of proofs:

```
start(+atom,+object_identifier) - one
```

start/3

Starts the MCP server with the given server name, application object, and options. Currently supported options: `server_version(Version)` to set the server version (default '1.0.0'), `server_title(Title)` to set the server display title (default '', omitted when empty).

Compilation flags:

`static`

Template:

`start(Name,Application,Options)`

Mode and number of proofs:

`start(+atom,+object__identifier,+list)` - one

start/4

Starts the MCP server with custom input and output streams. Useful for testing or non-stdio transports. Uses 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 the MCP server with custom input and output 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

elicit_request/5

Sends an elicitation/create request to the MCP client and reads the response. Input and Output are the I/O streams. Message is the prompt text (an atom). RequestedSchema is a curly-term JSON Schema describing the input to request from the user (e.g., {type-object, properties-{answer-{type-string, enum-[yes, no]}}, required-[answer]}). Answer is unified with accept(Content) (where Content is the user response as a curly-term), decline, or cancel. This predicate is typically not called directly but through the elicit closure passed to tool_call/4.

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

(no local declarations; see entity ancestors if any)

Private predicates

initialized_/0

Flag indicating that the server has been initialized.

Compilation flags:

dynamic

Mode and number of proofs:

initialized_ - zero_or_one

`application_/1`

Current application object.

Compilation flags:
dynamic

Template:
application_(Application)
Mode and number of proofs:
application_(?object_identifier) - zero_or_one

`server_name_/1`

Current server name.

Compilation flags:
dynamic

Template:
server_name_(Name)
Mode and number of proofs:
server_name_(?atom) - zero_or_one

`server_version_/1`

Current server version.

Compilation flags:
dynamic

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

`server_title_/1`

Current server display title.

Compilation flags:

`dynamic`

Template:

`server_title_(Title)`

Mode and number of proofs:

`server_title_(?atom) - zero_or_one`

`elicit_counter_/1`

Counter for elicitation request identifiers.

Compilation flags:

`dynamic`

Template:

`elicit_counter_(Counter)`

Mode and number of proofs:

`elicit_counter_(?integer) - zero_or_one`

`application_capabilities_/1`

Cached application capabilities list.

Compilation flags:

`dynamic`

Template:

`application_capabilities_(Capabilities)`

Mode and number of proofs:

`application_capabilities_(?list) - zero_or_one`

Operators

(none)

protocol

1.77.4 mcp_tool_protocol

Protocol for Logtalk objects that provide tools to be exposed via an MCP (Model Context Protocol) server. Implements the MCP 2025-06-18 specification. Implementing objects must define the set of tools available and handle tool calls. Tool metadata (names, titles, descriptions, parameter schemas) can be derived automatically from `info/2` and `mode/2` directives on the tool predicates, or can be specified explicitly via the `tool/1` predicate.

Availability:

```
logtalk_load(mcp_server(loader))
```

Author: Paulo Moura

Version: 0:3:0

Date: 2026-02-24

Compilation flags:

```
static
```

Dependencies:

(none)

Remarks:

- Capabilities: Objects can optionally define `capabilities/1` to declare which MCP capabilities they require. Currently supported: `elicitation` (allows the server to ask the user questions during tool execution via MCP elicitation). If `capabilities/1` is not defined, only the tools 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.
- Structured output: Tools can declare an output schema via `output_schema/2` and 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: 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`.

Inherited public predicates:

(none)

- Public predicates
 - capabilities/1
 - tools/1
 - tool_call/3
 - tool_call/4
 - output_schema/2
- Protected predicates
- Private predicates
- Operators

Public predicates

capabilities/1

Returns a list of MCP capabilities required by this tool provider. Currently supported capabilities: elicitation (server can ask the user questions during tool execution). If not defined, only the tools capability is advertised. The server uses this to build the capabilities field in the initialize response.

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. 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. Falls back to `tool_call/3` and then auto-dispatch if not defined.

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 is included in the tool descriptor as outputSchema and 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

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.78 memcached

object

1.78.1 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: 1:0:0

Date: 2026-02-09

Compilation flags:

```
static, context_switching_calls
```

Uses:

```
list
socket
user
```

Remarks:

- Supported backends: ECLiPSe, GNU Prolog, SICStus 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

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

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

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

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

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

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

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

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

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

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

`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`

`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`

`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

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

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

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.79 meta

object

1.79.1 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

1.79.2 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) - 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) - 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) - 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`

1.80 meta_compiler

object

1.80.1 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: 0:16:0

Date: 2024-10-24

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

1.81 metagol

object

1.81.1 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: 0:24:4

Date: 2024-03-15

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
- body_pred/1
- ibk/3
- func_test/3
- functional/0
- min_clauses/1
- max_clauses/1
- max_inv_preds/1
- metarule_next_id/1
- timeout/1
- Protected predicates
 - pprint_clause/1
 - pprint_clauses/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

Compilation flags:
static

head_pred/1

Compilation flags:
static

body_pred/1

Compilation flags:
dynamic

ibk/3

Compilation flags:
static

func_test/3

Compilation flags:
static

functional/0

Compilation flags:
dynamic

min_clauses/1

Compilation flags:
dynamic

max_clauses/1

Compilation flags:
dynamic

max_inv_preds/1

Compilation flags:
dynamic

metarule_next_id/1

Compilation flags:
dynamic

timeout/1

Compilation flags:
dynamic

Protected predicates

`pprint_clause/1`

Compilation flags:
static

`pprint_clauses/1`

Compilation flags:
static

`compiled_pred_call/2`

Compilation flags:
dynamic

`body_pred_call/2`

Compilation flags:
dynamic

`type/3`

Compilation flags:
dynamic

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.81.2 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)

1.82 mime_types

object

1.82.1 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: 1:0:0

Date: 2026-04-09

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(+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)

`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)

1.83 mutation_testing

object

1.83.1 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

1.83.2 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

1.83.3 `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

1.83.4 `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: 1:0:0

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

- 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

1.83.5 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

1.83.6 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

1.83.7 `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

1.83.8 `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

1.83.9 `mutation_testing`

Mutation testing tool.

Availability:

```
logtalk_load(mutation_testing(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-03

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

1.83.10 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

1.83.11 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

1.83.12 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

1.83.13 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

1.83.14 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

1.83.15 `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

1.83.16 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: 1:0:0

Date: 2026-03-07

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_identifer,?predicate_indicator,?list(integer),?integer) - zero_or_more

Operators

(none)

object

1.83.17 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: 1:0:0

Date: 2026-03-20

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

1.83.18 `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

1.83.19 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)

1.84 mutations

object

1.84.1 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

1.84.2 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

1.84.3 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

1.84.4 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

1.84.5 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

1.84.6 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

1.84.7 `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

1.85 naive_bayes

object

1.85.1 naive_bayes

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(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-20

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public classifier_protocol
```

Uses:

```
format
```

```
list
```

```
pairs
```

```
population
```

Remarks:

- Algorithm: Naive Bayes is a probabilistic classifier based on Bayes theorem with strong (naive) independence assumptions between features.
- Categorical features: Uses Laplace smoothing to handle unseen feature values.
- Continuous features: Uses Gaussian (normal) distribution to model numeric features.

- Classifier representation: The learned classifier is represented by default as `nb_classifier(Classes, ClassPriors, AttributeNames, FeatureTypes, FeatureParams)` where `FeatureParams` contains the learned probabilities or statistics for each feature.

Inherited public predicates:

`classifier_to_clauses/4 classifier_to_file/4 learn/2 predict/3 print_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. 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)`

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

`dataset_protocol`, `isolation_forest`, `c45`, `knn`, `nearest_centroid`, `random_forest`, `ada_boost`

1.86 nanoid

object

1.86.1 nanoid

NanoID generator using atom representation, 21 symbols, and the standard URL alphabet.

Availability:

`logtalk_load(nanoid(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

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), ids, ulid, uuid

object

1.86.2 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: 1:0:0

Date: 2026-02-26

Compilation flags:

static, context_switching_calls

Implements:

public nanoid_protocol

Uses:

fast_random(Algorithm)

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

nanoid, ids(Representation,Bytes), ulid(Representation), uuid(Representation)

protocol

1.86.3 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)

1.87 nearest_centroid

object

1.87.1 nearest_centroid

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(loader))`

Author: Paulo Moura

Version: 1:0:0
Date: 2026-02-20

Compilation flags:
static, context_switching_calls

Implements:
public classifier_protocol

Imports:
public options

Uses:
format
list
numberlist
pairs
type

Remarks:

- Algorithm: Assign to an instance the the class of the training samples whose mean (centroid) is closest to the instance.
- Distance metrics: Supports Euclidean, Manhattan, and cosine distance metrics.
- Feature types: Automatically handles numeric and categorical features.
- Classifier representation: The learned classifier is represented by default as a nc_classifier(AttributeNames, FeatureTypes, Centroids) term.

Inherited public predicates:

check_option/1 check_options/1 classifier_to_clauses/4 classifier_to_file/4 default_option/1
default_options/1 learn/2 option/2 option/3 predict/3 print_classifier/1 valid_option/1
valid_options/1

- Public predicates
 - predict/4
 - predict_probabilities/3
 - 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/3

Predicts class probabilities for a new instance using the learned classifier and default 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)

Mode and number of proofs:

predict_probabilities(+compound,+list,-list) - 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`, `c45`, `knn`, `naive_bayes`, `random_forest`, `ada_boost`

1.88 nested_dictionaries

object

1.88.1 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

1.88.2 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

1.88.3 nested_dictionary_protocol

Nested dictionary protocol.

Availability:

`logtalk_load(nested_dictionaries(loader))`

Author: Paul Brown and Paulo Moura

Version: 0:1:0

Date: 2021-04-07

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-bracketed term representation.

Compilation flags:

`static`

Template:

`as_nested_dictionary(Term,Dictionary)`

Mode and number of proofs:

`as_nested_dictionary(++term,--dictionary) - one_or_error`

`as_curly_bracketed/2`

Creates a a curly-bracketed 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`

`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

1.88.4 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

1.89 optionals

object

1.89.1 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

1.89.2 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

1.89.3 `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](#)

1.90 options

category

1.90.1 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: 1:2:0

Date: 2022-01-03

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

1.90.2 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](#)

1.91 os

object

1.91.1 os

Portable operating-system access predicates.

Availability:

`logtalk_load(os(loader))`

Author: Paulo Moura

Version: 1:103:1

Date: 2026-04-06

Compilation flags:

static, context_switching_calls

Implements:

public osp

Uses:

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 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

1.91.2 [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

1.91.3 osp

Portable operating-system access protocol.

Availability:

`logtalk_load(os(loader))`

Author: Paulo Moura

Version: 1:43:0

Date: 2026-04-03

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
- 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`

`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`

1.92 packs

`protocol`

1.92.1 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

1.92.2 packs

Pack handling predicates.

Availability:

`logtalk_load(packs(loader))`

Author: Paulo Moura

Version: 0:89:0

Date: 2026-03-24

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
- 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)

Remarks:

- lock(Boolean) option: Save lock extension facts (lockfile version, git registry commits, and pack integrity hashes) in addition to the standard requirements facts. 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)

```

Remarks:

- `lock(Boolean)` option: Require lock extension facts and enforce strict restoring (exact versions, git registry commits, and pack integrity hashes). 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. 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

(no local declarations; see entity ancestors if any)

Operators

(none)

category

1.92.3 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

1.92.4 `packs_messages`

Packs default message translations.

Availability:

`logtalk_load(packs(loader))`

Author: Paulo Moura

Version: 0:42:0

Date: 2026-03-02

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)

object

1.92.5 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

1.92.6 registries

Registry handling predicates.

Availability:

`logtalk_load(packs(loader))`

Author: Paulo Moura

Version: 0:64:1

Date: 2026-03-21

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

1.92.7 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

1.92.8 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)

1.93 pddl_parser

object

1.93.1 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

1.93.2 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)

1.94 permutations

object

1.94.1 permutations

Implementation of permutations operations over lists.

Availability:

`logtalk_load(permutations(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

Compilation flags:

static, context_switching_calls

Implements:

public permutations_protocol

Uses:

fast_random(Algorithm)

list

natural

Remarks:

(none)

Inherited public predicates:

cartesian_product/3 count_permutations/2 derangement/2 derangements/2
 distinct_permutation/2 distinct_permutation/3 distinct_permutations/2 distinct_permutations/3
 k_permutation/3 k_permutation/4 k_permutations/3 k_permutations/4 next_permutation/2
 nth_permutation/3 permutation/2 permutation/3 permutation_index/3 permutations/2
 permutations/3 previous_permutation/2 random_permutation/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

1.94.2 permutations_protocol

Protocol for permutations operations over lists.

Availability:

logtalk_load(permutations(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
 - 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
- cartesian_product/3
- derangements/2
- derangement/2
- next_permutation/2
- previous_permutation/2
- nth_permutation/3
- permutation_index/3
- count_permutations/2
- random_permutation/2
- Protected predicates
- Private predicates
- Operators

Public predicates

permutations/2

Generates all permutations of a list.

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.

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).

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.

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.

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.

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`

`cartesian_product/3`

Generates all K-element tuples from a list with replacement where order matters.

Compilation flags:

`static`

Template:

`cartesian_product(K,List,Tuples)`

Mode and number of proofs:

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

derangements/2

Generates all derangements of a list.

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.

Compilation flags:

static

Template:

derangement(List,Derangement)

Mode and number of proofs:

derangement(+list,-list) - one_or_more

next_permutation/2

Returns the next permutation 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 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.

Compilation flags:

`static`

Template:

`nth_permutation(List,Index,Permutation)`

Mode and number of proofs:

`nth_permutation(+list,+integer,-list) - zero_or_one`

`permutation_index/3`

Returns the zero-based index of a permutation.

Compilation flags:

`static`

Template:

`permutation_index(List,Permutation,Index)`

Mode and number of proofs:

`permutation_index(+list,+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`

`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`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.95 ports_profiler

object

1.95.1 ports_profiler

Predicate execution box model port profiler.

Availability:

`logtalk_load(ports_profiler(loader))`

Author: Paulo Moura

Version: 2:0:0

Date: 2024-05-18

Compilation flags:

`static, context_switching_calls`

Provides:

`logtalk::debug_handler/1`

`logtalk::debug_handler/3`

Uses:

`logtalk`

`user`

Remarks:

(none)

Inherited public predicates:

(none)

- 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)

1.96 process

object

1.96.1 process

Portable process handling predicates.

Availability:
logtalk_load(process(loader))

Author: Paulo Moura
Version: 1:0:1
Date: 2026-02-04

Compilation flags:
static, context_switching_calls

Dependencies:
(none)

Remarks:

- Supported backend Prolog systems: ECLiPSe, GNU Prolog, SICStus Prolog, SWI-Prolog, Trealla Prolog, and XVM.

Inherited public predicates:

(none)

- Public predicates
 - create/3
 - wait/2
 - 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), and stderr(Stream).

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.

Compilation flags:

static

Template:

wait(Process,Status)

Mode and number of proofs:

wait(+process_or_pid,-integer) - zero_or_one

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(Process,Signal)

Mode and number of proofs:

kill(+process_or_pid,+atom_or_integer) - zero_or_one

kill/1

Kills the given process using the default signal (sigkill).

Compilation flags:

static

Template:

kill(Process)

Mode and number of proofs:

kill(+process_or_pid) - zero_or_one

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

➡ See also

os

1.97 protobuf

object

1.97.1 protobuf

Google Protocol Buffers binary format parser and generator.

Availability:

`logtalk_load(protobuf(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-04

Compilation flags:

`static, context_switching_calls`

Uses:

`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

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

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

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

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.98 queues

object

1.98.1 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

1.98.2 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](#)

1.99 random

object

1.99.1 backend_random

Random number generator predicates using the backend Prolog compiler built-in random generator.

Availability:

`logtalk_load(random(loader))`

Author: Paulo Moura

Version: 1:22:0

Date: 2026-02-11

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` `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

1.99.2 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 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

1.99.3 `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: 3:0:0

Date: 2026-02-23

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 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

1.99.4 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 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

1.99.5 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 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

1.99.6 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: 3:0:0

Date: 2026-02-23

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 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 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

1.99.7 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: 3:3:0

Date: 2023-11-24

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
- 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

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 $0 \leq K \leq N$. 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

1.99.8 sampling_protocol

Predicates for sampling probability distributions.

Availability:

logtalk_load(random(loader))

Author: Paulo Moura

Version: 1:0:0

Date: 2025-02-25

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 $0.0 < \text{Shape} < 1$ and fails otherwise.

Compilation flags:

static

Template:

logseries(Shape,Value)

Mode and number of proofs:

logseries(+non_negative_integer,-positive_integer) - zero_or_one

geometric/2

Returns a geometric distributed random value (trials until the first success).

Compilation flags:

static

Template:

geometric(Probability,Value)

Mode and number of proofs:

geometric(+probability,-positive_integer) - one

hypergeometric/4

Returns a hypergeometric distributed random value.

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) - 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(+positive_integer,+positive_float,-float) - one

bernoulli/2

Returns a Bernoulli distributed random value.

Compilation flags:

static

Template:

bernoulli(Probability,Value)

Mode and number of proofs:

bernoulli(+positive_integer,-float) - 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(+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 integers 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. 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) - zero_or_one

`gumbel/3`

Returns a Gumbel distributed random value.

Compilation flags:
static

Template:
gumbel(Location,Scale,Value)
Mode and number of proofs:
gumbel(+float,+non_negative_float,-float) - zero_or_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(+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(+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,+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`

1.100 random_forest

object

1.100.1 random_forest

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(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-20

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public classifier_protocol
```

Imports:

```
public options
```

Uses:

```
c45
fast_random
format
list
pairs
type
```

Remarks:

- Algorithm: Random Forest is an ensemble learning method that constructs multiple decision trees during training and outputs the class that is the mode of the classes predicted by individual trees.
- Bootstrap sampling: Each tree is trained on a bootstrap sample (random sample with replacement) of the training data.
- Feature randomization: At each tree, a random subset of features is selected. The default number of features is `sqrt(total_features)`.
- Classifier representation: The learned classifier is represented by default as a `rf_classifier(Trees, ClassValues, Options)` term.

Inherited public predicates:

```
check_option/1 check_options/1 classifier_to_clauses/4 classifier_to_file/4 default_option/1
default_options/1 learn/2 option/2 option/3 predict/3 print_classifier/1 valid_option/1
valid_options/1
```

- Public predicates
 - `learn/3`
 - `predict_probabilities/3`
- 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
```

`predict_probabilities/3`

Predicts class probabilities for a new instance using the learned classifier. Returns a list of Class-Probability pairs sorted by descending probability. 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

`dataset_protocol`, `c45`, `isolation_forest`, `knn`, `naive_bayes`, `nearest_centroid`, `ada_boost`

1.101 reader

object

1.101.1 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: 2:2:0

Date: 2023-11-14

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
- 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`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

1.102 recorded_database

object

1.102.1 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

1.102.2 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)

1.103 redis

object

1.103.1 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)

1.104 sarif

object

1.104.1 sarif

Shared SARIF report generator for tools implementing the diagnostics protocol.

Availability:
logtalk_load(sarif(loader))

Author: Paulo Moura
Version: 1:0:0

Date: 2026-04-01

Compilation flags:

static, context__switching__calls

Uses:

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)

1.105 sbom

object

1.105.1 sbom

This tool generates a Software Bill of Materials (SBOM) for an application.

Availability:

```
logtalk_load(sbom(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-04-02

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 1.6 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 1.6 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)

1.106 sets

object

1.106.1 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

1.106.2 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,Lowes,Biggers)`

Mode and number of proofs:

`partition(+list,+nonvar,-list,-list) - one`

Operators

(none)

➡ See also

set, treap_set

protocol

1.106.3 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

1.106.4 `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)

1.107 simulated_annealing

object

1.107.1 simulated_annealing(Problem)

- Problem - Problem object implementing simulated_annealing_protocol.

Simulated annealing optimization algorithm using the Xoshiro128++ random number generator. Convenience object that extends simulated_annealing/2 with the random algorithm bound to xoshiro128pp.

Availability:

logtalk_load(simulated_annealing(loader))

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-22

Compilation flags:

static, context_switching_calls

Extends:

public simulated_annealing(Problem,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,RandomAlgorithm)`, `simulated_annealing_protocol`

object

1.107.2 `simulated_annealing(Problem,RandomAlgorithm)`

- `Problem` - Problem object implementing `simulated_annealing_protocol`.
- `RandomAlgorithm` - Random number generator algorithm for the `fast_random` library (e.g. `xoshiro128pp`, `xoshiro256ss`, `well512a`, ...).

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: 1:0:0

Date: 2026-02-23

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public options
```

Uses:

```
fast_random(Algorithm)  
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 $\exp(-\Delta E / \text{Temperature})$.
- Default cooling schedule: Geometric cooling: NewTemp is $\text{Temp} * 0.995$. 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

1.107.3 `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 re-computation. 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 ($\text{NewTemp} = \text{Temp} * 0.995$).

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)`

1.108 snowflakeid

object

1.108.1 snowflakeid

Snowflake ID generator using the Twitter-style profile and atom representation.

Availability:

```
logtalk_load(snowflakeid(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

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`

object

1.108.2 `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: 1:0:1

Date: 2026-03-13

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,`
`ksuid(Representation,Alphabet), cuid2(Representation,Size,Alphabet), nanoid(Representation,Size,Alphabet)`

object

1.108.3 snowflakeid_instagram

Instagram-style Snowflake profile using atom representation.

Availability:

`logtalk__load(snowflakeid(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

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

object

1.108.4 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: 1:0:1

Date: 2026-02-27

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`

protocol

1.108.5 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

1.108.6 snowflakeid_sonyflake

Sonyflake-style Snowflake profile using atom representation.

Availability:

logtalk_load(snowflakeid(loader))

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

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

object

1.108.7 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: 1:0:1

Date: 2026-02-27

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

object

1.108.8 snowflakeid_twitter

Twitter-style Snowflake profile using atom representation.

Availability:

```
logtalk_load(snowflakeid(loader))
```

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

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`

object

1.108.9 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: 1:0:1

Date: 2026-02-27

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`

1.109 sockets

object

1.109.1 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: 0:12:0

Date: 2026-04-01

Compilation flags:

static, context_switching_calls

Imports:

public options

Uses:

list

Remarks:

- Supported backends: ECLiPSe, GNU Prolog, SICStus Prolog, SWI-Prolog, and Trealla Prolog.
- 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`

`current_host/1`

Returns the hostname of the current machine.

Compilation flags:

`static`

Template:

`current_host(Host)`

Mode and number of proofs:

`current_host(-atom) - 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)

1.110 statistics

object

1.110.1 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

1.110.2 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

1.110.3 `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

1.110.4 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

1.111 stemming

object

1.1111.1 `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: 1:0:0

Date: 2026-02-05

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

1.111.2 `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

1.111.3 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)`

1.112 stomp

object

1.112.1 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: 1:0:0

Date: 2026-02-09

Compilation flags:

`static, context_switching_calls`

Uses:

`list`
`socket`
`term_io`
`user`
`uuid(Representation)`

Remarks:

- Supported backends: ECLiPSe, GNU Prolog, SICStus 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:

`(none)`

- 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`

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

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

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`

`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`

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

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

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`

`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`

`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`

`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`

`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)

1.113 string_distance

object

1.113.1 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: 1:0:0

Date: 2026-02-05

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
- metaphone/2
- metaphone_match/2
- double_metaphone/3
- double_metaphone_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 $1 - (\text{edit distance} / \text{max length of the two strings})$.

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 $1 - (\text{edit distance} / \text{max length of the two strings})$.

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 $|\text{intersection}| / |\text{union}|$.

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
```

[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

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.114 strings

object

1.114.1 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: 1:0:1

Date: 2026-03-07

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)

1.115 subsequences

object

1.115.1 subsequences

Implementation of subsequence operations over lists.

Availability:

`logtalk_load(subsequences(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

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

1.115.2 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)

1.116 term_io

object

1.116.1 `term_io`

Term input/output from/to atom, chars, and codes.

Availability:

`logtalk_load(term_io(loader))`

Author: Paulo Moura

Version: 1:3:0

Date: 2023-11-14

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`

1.116.2 `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: 1:3:0

Date: 2021-10-04

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`

`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`

`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`

`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`

`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`

`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`

`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`

`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`

`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`

`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

`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

`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
```

```
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
```

`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)

1.117 time_scales

object

1.117.1 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

1.117.2 `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: 0:2:0

Date: 2026-02-26

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

- 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(TAISecods,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

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.117.3 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`

1.118 timeout

object

1.118.1 timeout

Predicates for calling goal with a time limit.

Availability:

`logtalk_load(timeout(loader))`

Author: Paulo Moura

Version: 0:10:0

Date: 2022-06-15

Compilation flags:

static, context_switching_calls

Dependencies:

(none)

Remarks:

- Supported backend Prolog systems: B-Prolog, ECLiPSe, SICStus 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)

1.119 toml

object

1.119.1 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

1.119.2 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

1.119.3 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: 1:0:0

Date: 2026-04-09

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

1.119.4 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)

1.120 tool_diagnostics

category

1.120.1 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

1.120.2 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)

1.121 toon

object

1.121.1 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

1.121.2 `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

1.121.3 `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: 0:1:0

Date: 2025-12-07

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

1.121.4 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)

1.122 toychr

object

1.122.1 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: 0:7:2

Date: 2026-04-06

Copyright: Copyright 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`

Compilation flags:
static

`chr_trace/0`

Compilation flags:
static

`chr_notrace/0`

Compilation flags:
static

`chr_spy/1`

Compilation flags:
static

`chr_nospy/0`

Compilation flags:
static

chr_no_spy/1

Compilation flags:
static

chr_option/2

Compilation flags:
static

Protected predicates

current_prog/1

Compilation flags:
static

chr_option_print_trace/0

Compilation flags:
dynamic

chr_option_trace_interactive/0

Compilation flags:
dynamic

`chr_option_optimization_level/1`

Compilation flags:
dynamic

`chr_option_show_stack/0`

Compilation flags:
dynamic

`chr_option_show_store/0`

Compilation flags:
dynamic

`chr_option_show_history/0`

Compilation flags:
dynamic

`chr_option_show_id/0`

Compilation flags:
dynamic

chr_option_allow_deep_guards/0

Compilation flags:
dynamic

chr_next_state/1

Compilation flags:
dynamic

chr_spy_point/1

Compilation flags:
dynamic

Private predicates

chr_rule_/1

Compilation flags:
dynamic

Operators

(none)

1.123 tsv

object

1.123.1 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

1.123.2 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

1.123.3 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

1.123.4 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)

1.124 tutor

object

1.124.1 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`

1.124.2 `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: 0:87:0

Date: 2026-03-30

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)

1.125 types

object

1.125.1 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

1.125.2 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

1.125.3 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

1.125.4 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

1.125.5 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

1.125.6 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
- ($=\backslash=$)/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

$(=\backslash=)/2$

True if Term1 is not equal to Term2.

Compilation flags:

static

Template:

Term1= $\backslash=$ Term2

Mode and number of proofs:

+term= $\backslash=$ +term - zero_or_one

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

object

1.125.7 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

1.125.8 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

1.125.9 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 $\text{Lower} \leq \text{Upper}$; 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 $\text{Lower} \leq \text{Upper}$; 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

1.125.10 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

1.125.11 list

List predicates.

Availability:

logtalk_load(types(loader))

Author: Paulo Moura

Version: 4:4:0

Date: 2026-02-03

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

1.125.12 `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

1.125.13 listp

List protocol.

Availability:

logtalk_load(types(loader))

Author: Paulo Moura

Version: 1:19:0

Date: 2026-02-03

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. Fails if the list have fewer than N elements.

Compilation flags:

`static`

Template:

`take(N,List,Elements,Remaining)`

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

1.125.14 natural

Natural numbers data type predicates.

Availability:

logtalk_load(types(loader))

Author: Paulo Moura

Version: 2:0:0
Date: 2026-02-02

Compilation flags:
static, context_switching_calls

Extends:
public integer

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
 - binomial/3
- 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

`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`

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.125.15 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 $\text{abs}(\text{Number1} - \text{Number2}) \leq \max(\text{abs}(\text{Number1}), \text{abs}(\text{Number2})) * \text{epsilon}$. 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 $\text{abs}(\text{Number1} - \text{Number2}) \leq \max(\text{abs}(\text{Number1}), \text{abs}(\text{Number2})) * \text{Epsilon}$. 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 $\text{abs}(\text{Number1} - \text{Number2}) \leq \min(\text{abs}(\text{Number1}), \text{abs}(\text{Number2})) * \text{Epsilon}$. 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 $\text{abs}(\text{Number1} - \text{Number2}) \leq \max(\text{RelativeTolerance} * \max(\text{abs}(\text{Number1}), \text{abs}(\text{Number2})), \text{AbsoluteTolerance})$. 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

1.125.16 numberlist

List of numbers predicates.

Availability:

logtalk_load(types(loader))

Author: Paulo Moura

Version: 1:18:0

Date: 2026-02-23

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 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

1.125.17 numberlistp

List of numbers protocol.

Availability:

`logtalk_load(types(loader))`

Author: Paulo Moura

Version: 1:11:0

Date: 2026-02-23

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
- 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
```

`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 is 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 is 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 is 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 is 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.

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.

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

1.125.18 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

1.125.19 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 `term`

Aliases:

`term` 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

1.125.20 `term`

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 greater 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 greater 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

1.125.21 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: 2:8:0

Date: 2026-02-28

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`.
- 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)`.
- 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

1.125.22 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

1.125.23 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`

1.126 tzif

object

1.126.1 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: 1:0:0

Date: 2026-04-07

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`

1.126.2 `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: 1:0:0

Date: 2026-04-07

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

`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

`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`

`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`

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

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`

`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`

`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

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

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

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
```

```
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
```

```
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

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

`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
```

`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`

`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`

`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`

`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`

`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`

`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`

`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`

`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`

`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`

`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

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

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

object

1.126.3 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)

1.127 ulid

object

1.127.1 ulid

Universally Unique Lexicographically Sortable Identifier (ULID) generator using an atom representation.

Availability:

`logtalk_load(ulid(loader))`

Author: Paulo Moura

Version: 1:1:0

Date: 2026-02-26

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](#), [uuid](#)

object

1.127.2 [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: 1:1:0

Date: 2026-02-26

Compilation flags:

static, context_switching_calls

Implements:

public ulid_protocol

Uses:

fast_random

iso8601

list

os

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,Tim
uuid(Representation)

protocol

1.127.3 ulid_protocol

Universally Unique Lexicographically Sortable Identifier (ULID) generator protocol.

Availability:
 logtalk_load(ulid(loader))

Author: Paulo Moura
Version: 1:0:0
Date: 2023-05-17

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

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

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

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

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

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

category

1.127.4 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](#)

1.128 union_find

object

1.128.1 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

1.128.2 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](#)

1.129 url

object

1.129.1 url(Representation)

- Representation - URL and its 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: 1:1:0

Date: 2026-04-01

Compilation flags:

`static, context_switching_calls`

Uses:

[list](#)

Remarks:

(none)

Inherited public predicates:

(none)

- Public predicates
 - valid/1
 - parse/2
 - generate/2
 - normalize/2
 - file_path_components/2
- Protected predicates
- Private predicates
 - lowercase_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 from a list of its components: [scheme(Scheme), authority(Authority), path(Path), query(Query), fragment(Fragment)] for standard URLs, or scheme-specific components for mailto, news, tel, and urn URLs. Fails if the components are invalid.

Compilation flags:

```
static
```

Template:

```
generate(Components,URL)
```

Mode and number of proofs:

```
generate(++list(compound),-text) - zero_or_one
```

`normalize/2`

Normalizes a URL by standardizing its components. Normalization includes converting the scheme to lowercase, percent-encoding characters that require escaping, ensuring proper path separators, and handling relative paths.

Compilation flags:

```
static
```

Template:

```
normalize(URL,NormalizedURL)
```

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)

1.130 uuid

object

1.130.1 uuid

Universally unique identifier (UUID) generator using an atom representation.

Availability:

`logtalk_load(uuid(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-26

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_v3/3 uuid_v4/1
uuid_v5/3 uuid_v7/1`

- Public predicates
- Protected predicates
- Private predicates
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

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`, `ulid`

object

1.130.2 `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: 0:9:0

Date: 2026-04-08

Compilation flags:

`static`, `context_switching_calls`

Implements:

`public uuid_protocol`

Uses:

`fast_random`

`iso8601`

`list`

`os`

Remarks:

(none)

Inherited public predicates:

random_node/1 uuid_max/1 uuid_nil/1 uuid_null/1 uuid_v1/2 uuid_v3/3 uuid_v4/1
uuid_v5/3 uuid_v7/1

- Public predicates
- Protected predicates
- Private predicates
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

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,TimestampBi
ulid(Representation)

protocol

1.130.3 uuid_protocol

Universally unique identifier (UUID) generator protocol.

Availability:

logtalk_load(uuid(loader))

Author: Paulo Moura

Version: 0:6:0

Date: 2026-04-08

Compilation flags:

static

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- Public predicates
 - uuid_v1/2
 - uuid_v3/3
 - uuid_v4/1
 - uuid_v5/3
 - uuid_v7/1
 - 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.

Compilation flags:

`static`

Template:

`uuid_v1(MAC,UUID)`

Mode and number of proofs:

`uuid_v1(+list(byte),--text) - 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.

Compilation flags:

`static`

Template:

`uuid_v7(UUID)`

Mode and number of proofs:

`uuid_v7(--text) - 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)

1.131 validations

object

1.131.1 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

1.131.2 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: 1:0:0

Date: 2026-02-22

Compilation flags:

static, context__switching__calls

Provides:

type::type/1

type::check/2

Uses:

list

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

1.131.3 [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`

1.132 verdi_neruda

object

1.132.1 `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

1.132.2 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

1.132.3 `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

1.132.4 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

1.132.5 `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

1.132.6 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

1.132.7 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

1.132.8 `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

1.132.9 demodb

Availability:

`logtalk_load(verdi_neruda(loader))`

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

1.132.10 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

1.132.11 flattening

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

1.132.12 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

1.132.13 `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

1.132.14 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

1.132.15 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

1.132.16 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

1.132.17 `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 flattening
```

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

1.132.18 shell

User frontend to start the application.

Availability:

`logtalk_load(verdi_neruda(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2019-03-20

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

Compilation flags:

static

start/0

Compilation flags:

static

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.132.19 shell(Interpreters)

Prolog shell for the interpreters.

Availability:

logtalk_load(verdi_neruda(loader))

Author: Victor Lagerkvist and Paulo Moura

Version: 1:1:3

Date: 2024-03-15

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

Compilation flags:
static

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.132.20 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)

1.133 wrapper

object

1.133.1 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)

1.134 xml_parser

object

1.134.1 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
 - pcddata_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)

1.135 yaml

object

1.135.1 yaml

YAML parser and generator.

Availability:

`logtalk_load(yaml(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2026-02-03

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

1.135.2 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)

1.136 zippers

protocol

1.136.1 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

1.136.2 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)

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/`
- 2.2** `contributions/iso8601/`
- 2.3** `contributions/pddl_parser/`
- 2.4** `contributions/verdi_neruda/`
- 2.5** `contributions/xml_parser/`
- 2.6** `core/`
- 2.7** `library/`
- 2.8** `library/ada_boost/`
- 2.9** `library/amqp/`
- 2.10** `library/application/`
- 2.11** `library/arbitrary/`
- 2.12** `library/assignvars/`
- 2.13** `library/avro/`
- 2.14** `library/base32/`
- 2.15** `library/base58/`
- 2.16** `library/base64/`
- 2.17** `library/base85/`
- 2.18** `library/c45/`
- 2.19** `library/cbor/`
- 2.20** `library/ccsds/`
- 2.1.** `contributions/flags/`
- 2.21** `library/character_sets/`

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

3.2 Objects

3.3 Protocols

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.

4.1 `(/)/2`

- `help`

4.2 `(//)/2`

- `help`

4.3 `(<)/2`

- `comparingp`

4.4 `(<=)/2`

- `assignvarsp`
- `streamvars`

4.5 `(:=)/2`

- `comparingp`

4.6 $(=<)/2$

- `comparingp`

4.7 $(=>)/2$

- `assignvarsp`
- `streamvars`

4.8 $(=\backslash=)/2$

- `comparingp`

4.9 $=\sim= / 2$

- `lgtunit`
- `number`

4.10 $(>)/2$

- `comparingp`

4.11 $(>=)/2$

- `comparingp`

4.12 `abbreviation/2`

- `tzif_protocol`

4.13 `abbreviation/3`

- `tzif_protocol`

4.14 abbreviation/4

- tzif_protocol

4.15 abort_transaction/3

- stomp

4.16 absolute_file_name/2

- osp

4.17 ack/3

- stomp

4.18 acquire/1

- amqp_pool

4.19 activate_debug_handler/1

- logtalk

4.20 activate_monitor/0

- monitorp

4.21 active_debug_handler/1

- logtalk

4.22 add/1

- registries

4.23 add/2

- registries

4.24 add/3

- difflist
- registries

4.25 add/5

- memcached

4.26 add32/3

- hash_common_32

4.27 add32/4

- hash_common_32

4.28 add32/5

- hash_common_32

4.29 add64/3

- hash_common_64

4.30 addDependent/1

- subject

4.31 add_duration/3

- datep

4.32 add_edge/4

- unweighted_graph_protocol

4.33 add_edge/5

- weighted_graph_protocol

4.34 add_edges/3

- graph_protocol

4.35 add_rule/3

- datalog_protocol

4.36 add_type/2

- mime_types

4.37 add_type/3

- mime_types

4.38 add_vertex/3

- graph_protocol

4.39 add_vertices/3

- graph_protocol

4.40 after/2

- datep
- intervalp

4.41 after/3

- monitoring

4.42 alias/1

- character_set_protocol

4.43 all/0

- code_metric
- dead_code_scanner
- lgtdocp

4.44 all/1

- code_metric
- dead_code_scanner
- lgtdocp

4.45 all_files/0

- `diagram(Format)`
- `diagrams(Format)`

4.46 all_files/1

- `diagram(Format)`
- `diagrams(Format)`

4.47 all_libraries/0

- `diagram(Format)`
- `diagrams(Format)`

4.48 all_libraries/1

- `diagram(Format)`
- `diagrams(Format)`

4.49 all_pairs_min_paths/2

- `graph_protocol`

4.50 all_pairs_min_predecessors/2

- `graph_protocol`

4.51 all_score/1

- `code_metric`

4.52 along_track_distance/4

- `geospatial_protocol`

4.53 alternating_subsequence/2

- `subsequences_protocol`

4.54 alternating_subsequences/2

- `subsequences_protocol`

4.55 ancestor/1

- `hierarchyp`

4.56 ancestors/1

- `hierarchyp`

4.57 and64/3

- `hash_common_64`

4.58 apid/2

- `ccsds(SecondaryHeaderLength)`

4.59 apis/0

- `help`

4.60 apis/1

- help

4.61 append/2

- listp

4.62 append/3

- listp
- memcached
- queuep
- varlistp

4.63 append/4

- redis

4.64 apply/2

- zipperp

4.65 apply/4

- dictionaryp

4.66 approximately_equal/2

- lgtunit
- number

4.67 approximately_equal/3

- lgtunit
- number

4.68 arbitrary/1

- arbitrary

4.69 arbitrary/2

- arbitrary

4.70 archive/1

- registry_protocol

4.71 arithmetic_mean/2

- statisticsp

4.72 array_list/2

- java_utils_protocol

4.73 array_to_list/2

- java_utils_protocol

4.74 array_to_terms/2

- java_utils_protocol

4.75 array_to_terms/3

- java_utils_protocol

4.76 articulation_points/2

- undirected_graph_common

4.77 as_curly_bracketed/2

- dictionaryp
- nested_dictionary_protocol

4.78 as_deque/2

- deque_protocol

4.79 as_dictionary/2

- dictionaryp

4.80 as_difflist/2

- list

4.81 as_heap/2

- heap_protocol

4.82 as_list/2

- deque_protocol
- dictionaryp
- difflist
- heap_protocol
- queuep
- setp

4.83 `as__nested__dictionary/2`

- `nested__dictionary__protocol`

4.84 `as__set/2`

- `setp`

4.85 `ask__question/5`

- `logtalk`

4.86 `assert__fact/1`

- `datalog__protocol`

4.87 `assertion/1`

- `assertions(Mode)`
- `lgtunit`

4.88 `assertion/2`

- `assertions(Mode)`
- `lgtunit`

4.89 `assignable/1`

- `assignvarsp`

4.90 `assignable/2`

- `assignvarsp`

4.91 atom_string/2

- string(Representation)

4.92 atomics_to_string/2

- string(Representation)

4.93 atomics_to_string/3

- string(Representation)

4.94 attribute_values/2

- dataset_protocol

4.95 available/0

- packs

4.96 available/1

- packs

4.97 available/2

- packs

4.98 average/2

- numberlistp

4.99 average_deviation/3

- statisticsp

4.100 basic_ack/3

- amqp

4.101 basic_cancel/3

- amqp

4.102 basic_consume/3

- amqp

4.103 basic_get/3

- amqp

4.104 basic_nack/3

- amqp

4.105 basic_publish/4

- amqp

4.106 basic_qos/2

- amqp

4.107 basic__recover/2

- amqp

4.108 basic__reject/3

- amqp

4.109 bbox__contains/2

- geospatial_protocol

4.110 bbox__expand/3

- geospatial_protocol

4.111 bbox__from__coordinates/2

- geospatial_protocol

4.112 bbox__intersects/2

- geospatial_protocol

4.113 bbox__union/3

- geospatial_protocol

4.114 before/2

- datep
- intervalp

4.115 before/3

- monitoring

4.116 begin/0

- datalog_protocol

4.117 begin_transaction/3

- stomp

4.118 bench_goal/1

- databasep

4.119 benchmark/2

- lgtunit

4.120 benchmark/3

- lgtunit

4.121 benchmark/4

- lgtunit

4.122 benchmark_reified/3

- lgtunit

4.123 bernoulli/2

- `sampling_protocol`

4.124 beta/3

- `sampling_protocol`

4.125 between/3

- `integer`
- `random_protocol`

4.126 between/4

- `float`

4.127 big_endian_word32/2

- `hash_common_32`

4.128 binomial/3

- `natural`
- `sampling_protocol`

4.129 bit//1

- `number_grammars(Format)`

4.130 bits//1

- `number_grammars(Format)`

4.131 blank//0

- blank_grammars(Format)

4.132 blanks//0

- blank_grammars(Format)

4.133 body_pred/1

- metagol

4.134 bounding_box/3

- geospatial_protocol

4.135 branch/2

- git_protocol

4.136 breadth_first_order/3

- graph_protocol

4.137 bridges/2

- undirected_graph_common

4.138 built_date/1

- application_protocol

4.139 `built_in_directive/4`

- `help`

4.140 `built_in_flag/2`

- `flags`

4.141 `built_in_method/4`

- `help`

4.142 `built_in_non_terminal/4`

- `help`

4.143 `built_in_predicate/4`

- `help`

4.144 `bytes_hex/2`

- `hash_common_32`

4.145 `bytes_to_codes/2`

- `character_set_protocol`

4.146 `cache/1`

- `tzif_protocol`

4.147 cache_source/1

- tzif_protocol

4.148 cached_tzif/1

- tzif_protocol

4.149 calendar_month/3

- iso8601

4.150 call_with_timeout/2

- timeout

4.151 call_with_timeout/3

- timeout

4.152 capabilities/1

- mcp_tool_protocol

4.153 cartesian_product/3

- permutations_protocol

4.154 cas/6

- memcached

4.155 cat/2

- maybe

4.156 central_moment/3

- statisticsp

4.157 change_directory/1

- osp

4.158 changed/0

- subject

4.159 changed/1

- subject

4.160 channel_close/1

- amqp

4.161 channel_close/3

- amqp

4.162 channel_open/3

- amqp

4.163 `chebyshev_distance/3`

- `numberlistp`

4.164 `chebyshev_norm/2`

- `numberlistp`

4.165 `check/0`

- `command_line_option`

4.166 `check/1`

- `term p`
- `varlistp`

4.167 `check/2`

- `type`

4.168 `check/3`

- `type`

4.169 `check_conversion/3`

- `time_scales_protocol`

4.170 `check_convert/4`

- `time_scales_protocol`

4.171 check_instant/1

- time_scales_protocol

4.172 check_offset/3

- time_scales_protocol

4.173 check_option/1

- options_protocol

4.174 check_options/1

- options_protocol

4.175 chi_squared/2

- sampling_protocol

4.176 chr_is/2

- toychrdb

4.177 chr_no_spy/1

- toychrdb

4.178 chr_nospy/0

- toychrdb

4.179 chr_notrace/0

- toychrdb

4.180 chr_option/2

- toychrdb

4.181 chr_spy/1

- toychrdb

4.182 chr_trace/0

- toychrdb

4.183 circular_uniform_cartesian/3

- sampling_protocol

4.184 circular_uniform_polar/3

- sampling_protocol

4.185 class/1

- class_hierarchy
- dataset_protocol

4.186 class_values/1

- dataset_protocol

4.187 classes/1

- class_hierarchy

4.188 classifier_to_clauses/4

- classifier_protocol

4.189 classifier_to_file/4

- classifier_protocol

4.190 clause/5

- ports_profiler

4.191 clause_location/6

- ports_profiler

4.192 clean/0

- packs
- registries

4.193 clean/1

- packs
- registries

4.194 clean/2

- packs

4.195 clear/0

- `datalog_protocol`

4.196 clear__cache/0

- `tzif_protocol`

4.197 clear__dut1__override/0

- `time_scales_data`
- `time_scales_protocol`

4.198 clear__leap__seconds__override/0

- `time_scales_data`
- `time_scales_protocol`

4.199 client__open/4

- `socket`

4.200 client__open/5

- `socket`

4.201 clockwise__polygon/2

- `geospatial_protocol`

4.202 clone/1

- `cloning`
- `registry_protocol`

4.203 clone/3

- dictionaryp

4.204 clone/4

- dictionaryp

4.205 close/1

- amqp

4.206 close/2

- socket

4.207 close/3

- amqp

4.208 close_client/1

- linda_client

4.209 close_polygon/2

- geospatial_protocol

4.210 codes_to_bytes/2

- character_set_protocol

4.211 coefficient_of_variation/2

- statisticsp

4.212 combination/3

- combinations_protocol

4.213 combination/4

- combinations_protocol

4.214 combination_index/4

- combinations_protocol

4.215 combination_with_replacement/3

- combinations_protocol

4.216 combination_with_replacement/4

- combinations_protocol

4.217 combinations/3

- combinations_protocol

4.218 combinations/4

- combinations_protocol

4.219 combinations_with_replacement/3

- combinations_protocol

4.220 combinations_with_replacement/4

- combinations_protocol

4.221 command_line_arguments/1

- osp

4.222 commit/0

- datalog_protocol

4.223 commit_author/2

- git_protocol

4.224 commit_date/2

- git_protocol

4.225 commit_hash/2

- git_protocol

4.226 commit_hash_abbreviated/2

- git_protocol

4.227 commit_log/3

- git_protocol

4.228 commit_message/2

- git_protocol

4.229 commit_transaction/3

- stomp

4.230 common_subsequence/3

- subsequences_protocol

4.231 common_subsequences/3

- subsequences_protocol

4.232 compact/3

- json_ld_protocol

4.233 compare_date_time/3

- datep

4.234 compile_aux_clauses/1

- logtalk

4.235 compile_predicate_heads/4

- logtalk

4.236 compile_predicate_indicators/3

- logtalk

4.237 complement/2

- unweighted_graph_protocol

4.238 completion/2

- help

4.239 completions/2

- help

4.240 compose/3

- unweighted_directed_graph(Dictionary)

4.241 confirm_select/1

- amqp

4.242 connect/1

- redis

4.243 connect/2

- memcached

4.244 connect/3

- memcached
- redis

4.245 connect/4

- amqp
- stomp

4.246 connected_components/2

- unweighted_undirected_graph(Dictionary)
- weighted_undirected_graph(Dictionary)

4.247 connection_alive/1

- amqp
- stomp

4.248 console/1

- redis

4.249 contains/2

- intervalp

4.250 control//0

- blank_grammars(Format)

4.251 control_construct/4

- help

4.252 controls//0

- blank_grammars(Format)

4.253 convert/4

- time_scales_protocol

4.254 convert_zones/4

- dates_tz_protocol

4.255 convert_zones_with_resolution/5

- dates_tz_protocol

4.256 cooling_schedule/3

- simulated_annealing_protocol

4.257 coordinates_bounding_box/2

- geospatial_protocol

4.258 copy_file/2

- osp

4.259 correlation/3

- statisticsp

4.260 cosine_similarity/3

- string_distance(Representation)

4.261 count_combinations/3

- combinations_protocol

4.262 count_combinations_with_replacement/3

- combinations_protocol

4.263 count_distinct_subsequences/3

- subsequences_protocol

4.264 count_permutations/2

- permutations_protocol

4.265 count_subsequences/2

- subsequences_protocol

4.266 counter/2

- counters
- mutations_store

4.267 counterclockwise_polygon/2

- geospatial_protocol

4.268 covariance/3

- statisticsp

4.269 cover/1

- lgtunit

4.270 coverage_clause_mutator/0

- mutator_protocol

4.271 cpu_time/1

- osp
- timep

4.272 create/3

- process

4.273 creators/1

- application_protocol

4.274 cross_track_distance/4

- `geospatial_protocol`

4.275 current/2

- `zipperp`

4.276 current_host/1

- `socket`

4.277 cycle/2

- `directed_graph_protocol`
- `undirected_graph_common`

4.278 damerau_levenshtein/3

- `string_distance(Representation)`

4.279 data/0

- `ports_profiler`

4.280 data/1

- `ports_profiler`

4.281 data/2

- `ports_profiler`

4.282 data_length/2

- ccstds(SecondaryHeaderLength)

4.283 date/4

- iso8601

4.284 date/5

- iso8601

4.285 date/6

- iso8601

4.286 date/7

- iso8601

4.287 date_string/3

- iso8601

4.288 date_time/7

- osp

4.289 date_time_string/3

- iso8601

4.290 date_time_to_unix/2

- datep

4.291 day_of_year/2

- datep

4.292 day_of_year_date/3

- datep

4.293 daylight_saving_time/2

- tzif_protocol

4.294 daylight_saving_time/3

- tzif_protocol

4.295 daylight_saving_time/4

- tzif_protocol

4.296 days_in_month/3

- datep

4.297 deactivate_debug_handler/0

- logtalk

4.298 debug/0

- debuggerp

4.299 debug_handler/1

- logtalk

4.300 debug_handler/3

- logtalk

4.301 debugging/0

- debuggerp

4.302 debugging/1

- debuggerp

4.303 decide/1

- fcube

4.304 decide/2

- fcube

4.305 decode/2

- json_rpc

4.306 decode__exception/2

- java__utils__protocol

4.307 decode__exception/3

- java__utils__protocol

4.308 decode__frame/2

- amqp

4.309 decompile__predicate__heads/4

- logtalk

4.310 decompile__predicate__indicators/4

- logtalk

4.311 decompose__file__name/3

- osp

4.312 decompose__file__name/4

- osp

4.313 decr/3

- redis

4.314 `decr/4`

- `memcached`

4.315 `decrby/4`

- `redis`

4.316 `decrement_counter/1`

- `counters`

4.317 `default/1`

- `command_line_option`

4.318 `default_option/1`

- `options_protocol`
- `wrapper`

4.319 `default_options/1`

- `options_protocol`
- `wrapper`

4.320 `define_log_file/2`

- `loggingp`

4.321 `defined/4`

- `registries`

4.322 defined_flag/6

- flags

4.323 degree/3

- unweighted_undirected_graph(Dictionary)
- weighted_undirected_graph(Dictionary)

4.324 del/3

- redis

4.325 del_monitors/0

- event_registry

4.326 del_monitors/4

- event_registry

4.327 del_spy_points/4

- monitor

4.328 delete/0

- registries

4.329 delete/1

- registries

4.330 delete/2

- memcached
- registries

4.331 delete/3

- listp
- setp
- varlistp

4.332 delete/4

- dictionaryp
- heap_protocol

4.333 delete_all_after/2

- zipperp

4.334 delete_all_after_and_unzip/2

- zipperp

4.335 delete_all_before/2

- zipperp

4.336 delete_all_before_and_unzip/2

- zipperp

4.337 delete__and__next/2

- zipperp

4.338 delete__and__previous/2

- zipperp

4.339 delete__and__unzip/2

- zipperp

4.340 delete__directory/1

- osp

4.341 delete__directory__and__contents/1

- osp

4.342 delete__directory__contents/1

- osp

4.343 delete__edge/4

- unweighted_graph_protocol

4.344 delete__edge/5

- weighted_graph_protocol

4.345 delete__edges/3

- graph_protocol

4.346 delete__file/1

- osp

4.347 delete__in/4

- nested_dictionary_protocol

4.348 delete__matches/3

- listp

4.349 delete__max/4

- dictionaryp

4.350 delete__min/4

- dictionaryp

4.351 delete__vertex/3

- graph_protocol

4.352 delete__vertices/3

- graph_protocol

4.353 depends/1

- packs
- subject

4.354 depends/2

- packs

4.355 depends/3

- packs

4.356 depth/2

- term

4.357 depth_first_order/3

- graph_protocol

4.358 derangement/2

- permutations_protocol

4.359 derangements/2

- permutations_protocol

4.360 descendant/1

- hierarchyp

4.361 descendant_class/1

- class_hierarchy

4.362 descendant_classes/1

- class_hierarchy

4.363 descendant_instance/1

- class_hierarchy

4.364 descendant_instances/1

- class_hierarchy

4.365 descendants/1

- hierarchy

4.366 describe/1

- packs
- registries

4.367 describe/2

- packs

4.368 description/1

- application_protocol
- pack_protocol
- registry_protocol

4.369 destination_point/4

- geospatial_protocol

4.370 destroy/0

- amqp_pool

4.371 deterministic/1

- lgtunit

4.372 deterministic/2

- lgtunit

4.373 diagnostic/2

- tool_diagnostics_protocol

4.374 diagnostic/3

- tool_diagnostics_protocol

4.375 diagnostic_rule/5

- tool_diagnostics_protocol

4.376 diagnostic_rules/1

- tool_diagnostics_protocol

4.377 diagnostic_target/1

- tool_diagnostics_protocol

4.378 diagnostics/2

- tool_diagnostics_protocol

4.379 diagnostics/3

- tool_diagnostics_protocol

4.380 diagnostics_preflight/2

- tool_diagnostics_protocol

4.381 diagnostics_preflight/3

- tool_diagnostics_protocol

4.382 diagnostics_summary/2

- tool_diagnostics_protocol

4.383 diagnostics_summary/3

- tool_diagnostics_protocol

4.384 diagnostics_tool/5

- tool_diagnostics_protocol

4.385 diagram_description/1

- diagram(Format)

4.386 diagram_name_suffix/1

- diagram(Format)

4.387 dif/1

- coroutining
- dif

4.388 dif/2

- coroutining
- dif

4.389 digit//1

- number_grammars(Format)

4.390 digits//1

- number_grammars(Format)

4.391 directories/1

- lgtdocp
- wrapper

4.392 directories/2

- `diagram(Format)`
- `diagrams(Format)`
- `lgtdocp`
- `wrapper`

4.393 directories/3

- `diagram(Format)`
- `diagrams(Format)`

4.394 directory/1

- `code__metric`
- `dead__code__scanner`
- `diagram(Format)`
- `diagrams(Format)`
- `lgtdocp`
- `mutation__testing`
- `packs__common`
- `wrapper`

4.395 directory/2

- `code__metric`
- `dead__code__scanner`
- `diagram(Format)`
- `diagrams(Format)`
- `lgtdocp`
- `mutation__testing`
- `packs__common`
- `wrapper`

4.396 directory/3

- `directory(Format)`
- `directories(Format)`

4.397 directory_exists/1

- `directory_exists`

4.398 directory_files/2

- `directory_files`

4.399 directory_files/3

- `directory_files`

4.400 directory_mutants/2

- `directory_mutants`

4.401 directory_mutants/3

- `directory_mutants`

4.402 directory_score/2

- `directory_score`

4.403 dirichlet/2

- `dirichlet_protocol`

4.404 disable/0

- `linter_reporter`

4.405 disable/1

- `debug_messages`

4.406 disable/2

- `debug_messages`

4.407 disable_logging/1

- `loggingp`

4.408 disconnect/1

- `memcached`
- `redis`

4.409 disconnect/2

- `stomp`

4.410 disjoint/2

- `setp`

4.411 disjoint_sets/2

- `union_find_protocol`

4.412 distance/4

- `geospatial`
- `geospatial_protocol`

4.413 distance/5

- `geospatial`
- `geospatial_protocol`

4.414 distinct_combination/3

- `combinations_protocol`

4.415 distinct_combination/4

- `combinations_protocol`

4.416 distinct_combinations/3

- `combinations_protocol`

4.417 distinct_combinations/4

- `combinations_protocol`

4.418 distinct_permutation/2

- `permutations_protocol`

4.419 distinct_permutation/3

- `permutations_protocol`

4.420 `distinct_permutations/2`

- `permutations_protocol`

4.421 `distinct_permutations/3`

- `permutations_protocol`

4.422 `distribution/1`

- `application_protocol`

4.423 `doc_goal/1`

- `doclet`

4.424 `document/1`

- `sbom`

4.425 `document/2`

- `sbom`

4.426 `dot//1`

- `number_grammars(Format)`

4.427 `double_metaphone/3`

- `string_distance(Representation)`

4.428 double_metaphone_match/2

- string_distance(Representation)

4.429 dowhile/2

- loopp

4.430 drop/3

- listp

4.431 duration_between/3

- datep

4.432 duration_string/2

- iso8601

4.433 during/2

- intervalp

4.434 dut1_entries/1

- time_scales_data
- time_scales_protocol

4.435 dut1_offset_at_utc_unix/3

- time_scales_data

4.436 dut1_source/1

- time_scales_data
- time_scales_protocol

4.437 easter_day/3

- iso8601

4.438 edge/3

- unweighted_graph_protocol

4.439 edge/4

- weighted_graph_protocol

4.440 edge/6

- graph_language_protocol

4.441 edge_case/2

- arbitrary

4.442 edges/2

- graph_protocol

4.443 edit_similarity/3

- string_distance(Representation)

4.444 edit_similarity/4

- string_distance(Representation)

4.445 either/3

- expected(Expected)

4.446 elicit_request/5

- mcp_server

4.447 empty/1

- deque_protocol
- dictionaryp
- graph_protocol
- heap_protocol
- listp
- nested_dictionary_protocol
- optional
- queuep
- setp
- varlistp

4.448 enable/0

- linter_reporter

4.449 enable/1

- debug_messages
- linter_reporter

4.450 enable/2

- debug_messages

4.451 enable_logging/1

- loggingp

4.452 enabled/1

- debug_messages

4.453 enabled/2

- debug_messages

4.454 encode/2

- json_rpc

4.455 encode_frame/2

- amqp

4.456 encoding_suffix/2

- mime_types

4.457 ensure_directory/1

- osp

4.458 ensure_file/1

- osp

4.459 entity/1

- code_metric
- dead_code_scanner
- help
- mutation_testing
- xref_diagram(Format)

4.460 entity/2

- dead_code_scanner
- mutation_testing
- xref_diagram(Format)

4.461 entity_info_pair_score_hook/3

- doc_metric

4.462 entity_info_score_hook/2

- doc_metric

4.463 entity_mutants/2

- mutation_testing

4.464 entity_mutants/3

- mutation_testing

4.465 entity_predicates_weights_hook/2

- doc_metric

4.466 entity_prefix/2

- logtalk

4.467 entity_score/2

- code_metric

4.468 enumerate/2

- random_protocol

4.469 environment_variable/2

- osp

4.470 epsilon/1

- lgtunit

4.471 equal/2

- intervalp
- setp

4.472 equirectangular_inverse/4

- geospatial_protocol

4.473 `equirectangular_projection/4`

- `geospatial_protocol`

4.474 `erase/1`

- `recorded_database_core`

4.475 `error/2`

- `json_rpc`

4.476 `error_code/2`

- `json_rpc`

4.477 `error_data/2`

- `json_rpc`

4.478 `error_message/2`

- `json_rpc`

4.479 `error_response/4`

- `json_rpc`

4.480 `error_response/5`

- `json_rpc`

4.481 essentially_equal/3

- lgtunit
- number

4.482 estimate_temperature/1

- simulated_annealing(Problem,RandomAlgorithm)

4.483 estimate_temperature/2

- simulated_annealing(Problem,RandomAlgorithm)

4.484 euclidean_distance/3

- numberlistp

4.485 euclidean_norm/2

- numberlistp

4.486 example/3

- dataset_protocol

4.487 exchange_bind/4

- amqp

4.488 exchange_declare/3

- amqp

4.489 exchange_delete/3

- amqp

4.490 exchange_unbind/4

- amqp

4.491 exclude/3

- metap

4.492 execution_context/7

- logtalk

4.493 exists/3

- redis

4.494 expand/2

- json_ld_protocol

4.495 expand_library_path/2

- logtalk

4.496 expected/1

- expected(Expected)

4.497 expecteds/2

- either

4.498 expire/4

- redis

4.499 explain/2

- datalog_protocol

4.500 explain//1

- tutor_explanations

4.501 exponential/2

- sampling_protocol

4.502 export/1

- sbom

4.503 export/2

- sbom

4.504 extension/1

- proto_hierarchy

4.505 `extension_type/2`

- `mime_types`

4.506 `extension_type/3`

- `mime_types`

4.507 `extensions/1`

- `proto_hierarchy`

4.508 `external_reference/2`

- `application_protocol`

4.509 `factorial/2`

- `natural`

4.510 `facts/1`

- `datalog_protocol`

4.511 `failed_test_reason//1`

- `lgtunit_messages`

4.512 `false/1`

- `java_utils_protocol`

4.513 fcube/0

- fcube

4.514 file/1

- code__metric
- dead__code__scanner
- entity__diagram(Format)
- lgtdocp
- wrapper

4.515 file/2

- code__metric
- dead__code__scanner
- entity__diagram(Format)
- lgtdocp
- wrapper

4.516 file__exists/1

- osp

4.517 file__footer/3

- graph__language__protocol

4.518 file__header/3

- graph__language__protocol

4.519 file__modification__time/2

- osp

4.520 file__path__components/2

- url(Representation)

4.521 file__permission/2

- osp

4.522 file__score/2

- code__metric

4.523 file__size/2

- osp

4.524 file__to__bytes/2

- reader

4.525 file__to__bytes/3

- reader

4.526 file__to__chars/2

- reader

4.527 file_to_chars/3

- reader

4.528 file_to_codes/2

- reader

4.529 file_to_codes/3

- reader

4.530 file_to_terms/2

- reader

4.531 file_to_terms/3

- reader

4.532 file_type_extension/2

- logtalk

4.533 files/1

- diagram(Format)
- diagrams(Format)
- lgtdocp
- wrapper

4.534 files/2

- `diagram(Format)`
- `diagrams(Format)`
- `lgtdocp`
- `wrapper`

4.535 files/3

- `diagram(Format)`
- `diagrams(Format)`

4.536 filter/2

- `optional(Optional)`

4.537 filter/3

- `expected(Expected)`
- `validation(Validation)`

4.538 final_bearing/3

- `geospatial_protocol`

4.539 find/4

- `union_find_protocol`

4.540 find/5

- `union_find_protocol`

4.541 findall_in_noblock/3

- linda_client

4.542 findall_in_noblock/4

- linda_client

4.543 findall_member/4

- metap

4.544 findall_member/5

- metap

4.545 findall_rd_noblock/3

- linda_client

4.546 findall_rd_noblock/4

- linda_client

4.547 finished_by/2

- intervalp

4.548 finishes/2

- intervalp

4.549 fisher/3

- `sampling_protocol`

4.550 flag_group_chk/1

- `flags`

4.551 flag_groups/1

- `flags`

4.552 flat_map/2

- `expected(Expected)`
- `optional(Optional)`
- `validation(Validation)`

4.553 flatten/1

- `expected(Expected)`
- `optional(Optional)`
- `validation(Validation)`

4.554 flatten/2

- `json_ld_protocol`
- `listp`
- `varlistp`

4.555 float//1

- `number_grammars(Format)`

4.556 flush_all/1

- memcached

4.557 flush_all/2

- memcached

4.558 fold_left/4

- metap

4.559 fold_left_1/3

- metap

4.560 fold_right/4

- metap

4.561 fold_right_1/3

- metap

4.562 fordownto/3

- looppp

4.563 fordownto/4

- looppp

4.564 fordownto/5

- looppp

4.565 foreach/3

- looppp

4.566 foreach/4

- looppp

4.567 format/2

- format

4.568 format/3

- format

4.569 format_date_time/4

- datep

4.570 format_entity_score//2

- code_metric

4.571 format_object/1

- diagram(Format)

4.572 `format_report/1`

- `mutation_testing`

4.573 `format_report/2`

- `mutation_testing`

4.574 `format_report/3`

- `mutation_testing`

4.575 `format_to_atom/3`

- `term_io_protocol`

4.576 `format_to_chars/3`

- `term_io_protocol`

4.577 `format_to_chars/4`

- `term_io_protocol`

4.578 `format_to_codes/3`

- `term_io_protocol`

4.579 `format_to_codes/4`

- `term_io_protocol`

4.580 `forto/3`

- `loopp`

4.581 `forto/4`

- `loopp`

4.582 `forto/5`

- `loopp`

4.583 `forward/1`

- `forwarding`

4.584 `forward/2`

- `zipperp`

4.585 `forward/3`

- `zipperp`

4.586 `fractile/3`

- `statisticsp`

4.587 `frame_body/2`

- `stomp`

4.588 frame_command/2

- stomp

4.589 frame_header/3

- stomp

4.590 frame_headers/2

- stomp

4.591 freeze/2

- coroutining

4.592 frequency_distribution/2

- statisticsp

4.593 from_expected/2

- validation

4.594 from_generator/2

- expected
- optional
- validation

4.595 from_generator/3

- expected
- optional
- validation

4.596 from_generator/4

- expected
- validation

4.597 from_goal/2

- expected
- optional
- validation

4.598 from_goal/3

- expected
- optional
- validation

4.599 from_goal/4

- expected
- validation

4.600 from_goal_or_throw/2

- optional

4.601 from_goal_or_throw/3

- optional

4.602 from_optional/3

- expected
- validation

4.603 frozen/2

- coroutining

4.604 full_device_path/1

- osp

4.605 func_test/3

- metagol

4.606 functional/0

- metagol

4.607 gamma/3

- sampling_protocol

4.608 gat/4

- memcached

4.609 gats/5

- memcached

4.610 generate/1

- cuid2_protocol
- ids(Representation,Bytes)
- ksuid_protocol
- nanoid_protocol
- snowflakeid_protocol
- ulid_protocol

4.611 generate/2

- base32
- base58
- base64
- base64url
- base85
- cbor(StringRepresentation)
- ccstds(SecondaryHeaderLength)
- html
- json_ld_protocol
- json_lines_protocol
- json_protocol
- sarif
- toml_protocol
- toon_protocol
- ulid_protocol
- url(Representation)
- yaml_protocol

4.612 generate/3

- avro
- ccstds(SecondaryHeaderLength)
- protobuf

4.613 generate/4

- avro
- protobuf
- sarif

4.614 generate/8

- ulid_protocol

4.615 generate_all/2

- yaml_protocol

4.616 genint/2

- genint_core

4.617 gensym/2

- gensym_core

4.618 geometric/2

- sampling_protocol

4.619 geometric_mean/2

- statisticsp

4.620 get/1

- optional(Optional)

4.621 get/3

- memcached
- redis

4.622 get/4

- memcached

4.623 get__field/2

- java__access__protocol

4.624 get__flag__value/2

- flags

4.625 get__seed/1

- arbitrary
- pseudo_random_protocol

4.626 getrange/5

- redis

4.627 gets/4

- memcached

4.628 gets/5

- memcached

4.629 git_object_identifer/1

- application_protocol

4.630 gnu/0

- fcube

4.631 goal_expansion/2

- expanding

4.632 graph_coloring/3

- undirected_graph_common

4.633 graph_footer/5

- graph_language_protocol

4.634 graph_header/5

- graph_language_protocol

4.635 ground/1

- term_p

4.636 group_by_key/2

- pairs

4.637 group_consecutive_by_key/2

- pairs

4.638 group_sorted_by_key/2

- pairs

4.639 guess_all_extensions/2

- mime_types

4.640 guess_all_extensions/3

- mime_types

4.641 guess_arity/2

- csv_protocol

4.642 guess_extension/2

- mime_types

4.643 guess_extension/3

- mime_types

4.644 guess_file_type/3

- mime_types

4.645 guess_file_type/4

- mime_types

4.646 guess_separator/2

- csv_protocol

4.647 guess_type/3

- mime_types

4.648 guess_type/4

- mime_types

4.649 gumbel/3

- sampling_protocol

4.650 hamming/3

- string_distance(Representation)

4.651 hamming_distance/3

- listp

4.652 handbook/0

- help

4.653 harmonic_mean/2

- statisticsp

4.654 has__cycle/1

- directed_graph_protocol
- undirected_graph_common

4.655 has__negative_cycle/1

- weighted_graph_protocol

4.656 has__path/3

- graph_protocol

4.657 hash/2

- hash_protocol

4.658 haversine_distance/3

- geospatial_protocol

4.659 hdel/4

- redis

4.660 head/2

- queuep

4.661 head__pred/1

- metagol

4.662 help/0

- help
- packs_common

4.663 help/1

- command_line_option

4.664 help/2

- command_line_options

4.665 help/3

- command_line_options

4.666 hex_digit//1

- number_grammars(Format)

4.667 hex_digits//1

- number_grammars(Format)

4.668 hexists/4

- redis

4.669 hget/4

- redis

4.670 hgetall/3

- redis

4.671 hkeys/3

- redis

4.672 hlen/3

- redis

4.673 home/1

- pack_protocol
- registry_protocol

4.674 homepage/1

- application_protocol

4.675 hset/5

- redis

4.676 hvals/3

- redis

4.677 hypergeometric/4

- sampling_protocol

4.678 ibk/3

- metagol

4.679 id/2

- json_rpc

4.680 if_empty/1

- optional(Optional)

4.681 if_expected/1

- expected(Expected)

4.682 if_expected_or_else/2

- expected(Expected)

4.683 if_invalid/1

- validation(Validation)

4.684 if_present/1

- optional(Optional)

4.685 if_present_or_else/2

- optional(Optional)

4.686 if_unexpected/1

- expected(Expected)

4.687 if_valid/1

- validation(Validation)

4.688 if_valid_or_else/2

- validation(Validation)

4.689 in/1

- linda_client

4.690 in/2

- linda_client

4.691 in_degree/3

- directed_graph_protocol

4.692 in_list/2

- linda_client

4.693 in_list/3

- linda_client

4.694 in_noblock/1

- linda_client

4.695 in_noblock/2

- linda_client

4.696 include/3

- metap

4.697 incr/3

- redis

4.698 incr/4

- memcached

4.699 incrby/4

- redis

4.700 increase/1

- counter

4.701 increment/0

- counter

4.702 increment_counter/1

- counters

4.703 init/0

- shell(Interpreters)

4.704 init/2

- subsequences_protocol

4.705 init1/2

- subsequences_protocol

4.706 init_log_file/2

- loggingp

4.707 init_tail/2

- subsequences_protocol

4.708 init_tails/2

- subsequences_protocol

4.709 initial_bearing/3

- geospatial_protocol

4.710 initial_state/1

- simulated_annealing_protocol

4.711 initial_temperature/1

- simulated_annealing_protocol

4.712 initialize/1

- amqp_pool

4.713 inits/2

- subsequences_protocol

4.714 inits1/2

- subsequences_protocol

4.715 inorder/2

- bintree

4.716 insert/3

- setp

4.717 insert/4

- dictionaryp
- heap_protocol

4.718 insert_after/3

- zipperp

4.719 insert_all/3

- heap_protocol
- setp

4.720 insert_before/3

- zipperp

4.721 insert_in/4

- nested_dictionary_protocol

4.722 install/1

- packs

4.723 install/2

- packs

4.724 install/3

- packs

4.725 install/4

- packs

4.726 installed/0

- packs

4.727 installed/1

- packs

4.728 installed/3

- packs

4.729 installed/4

- packs

4.730 instance/1

- class_hierarchy

4.731 instance/2

- recorded_database_core

4.732 instances/1

- class_hierarchy

4.733 instant_to_utc_date_time/2

- time_scales_protocol

4.734 integer//1

- number_grammars(Format)

4.735 integer_to_big_endian_bytes32/2

- hash_common_32

4.736 integer_to_big_endian_bytes64/2

- hash_common_64

4.737 integer_to_little_endian_bytes32/2

- hash_common_32

4.738 internal_error/2

- json_rpc

4.739 internal_os_path/2

- osp

4.740 interpolate_great_circle/4

- geospatial_protocol

4.741 interpolate_rhumb/4

- geospatial_protocol

4.742 `interquartile_range/2`

- `statisticsp`

4.743 `intersect/2`

- `setp`

4.744 `intersection/2`

- `dictionaryp`

4.745 `intersection/3`

- `dictionaryp`
- `setp`

4.746 `intersection/4`

- `setp`

4.747 `interval_string/2`

- `iso8601`

4.748 `invalid/1`

- `validation(Validation)`

4.749 `invalid_params/2`

- `json_rpc`

4.750 invalid_request/1

- json_rpc

4.751 invalids/2

- validated

4.752 invoke/1

- java_access_protocol

4.753 invoke/2

- java_access_protocol

4.754 ipv4//1

- ip_grammars(Format)

4.755 ipv6//1

- ip_grammars(Format)

4.756 is_absolute_file_name/1

- osp

4.757 is_acyclic/1

- directed_graph_protocol

4.758 is_alpha/1

- characterp

4.759 is_alphanumeric/1

- characterp

4.760 is_ascii/1

- characterp

4.761 is_batch/1

- json_rpc

4.762 is_bin_digit/1

- characterp

4.763 is_bipartite/1

- graph_protocol

4.764 is_clockwise_polygon/1

- geospatial_protocol

4.765 is_complete/1

- graph_protocol

4.766 is_connected/1

- unweighted_undirected_graph(Dictionary)
- weighted_undirected_graph(Dictionary)

4.767 is_control/1

- characterp

4.768 is_dec_digit/1

- characterp

4.769 is_empty/0

- optional(Optional)

4.770 is_end_of_line/1

- characterp

4.771 is_error_response/1

- json_rpc

4.772 is_expected/0

- expected(Expected)

4.773 is_false/1

- java_utils_protocol

4.774 is_hex_digit/1

- characterp

4.775 is_invalid/0

- validation(Validation)

4.776 is_layout/1

- characterp

4.777 is_letter/1

- characterp

4.778 is_lower_case/1

- characterp

4.779 is_newline/1

- characterp

4.780 is_notification/1

- json_rpc

4.781 is_null/1

- java_utils_protocol

4.782 is__object/1

- java_utils_protocol

4.783 is__octal_digit/1

- characterp

4.784 is__period/1

- characterp

4.785 is__prefix_of/2

- subsequences_protocol

4.786 is__present/0

- optional(Optional)

4.787 is__punctuation/1

- characterp

4.788 is__quote/1

- characterp

4.789 is__request/1

- json_rpc

4.790 is__response/1

- json_rpc

4.791 is__sparse/1

- graph_protocol

4.792 is__subsequence_of/2

- subsequences_protocol

4.793 is__suffix_of/2

- subsequences_protocol

4.794 is__tree/1

- undirected_graph_common

4.795 is__true/1

- java_utils_protocol

4.796 is__unexpected/0

- expected(Expected)

4.797 is__upper_case/1

- characterp

4.798 `is__valid/0`

- `validation(Validation)`

4.799 `is__valid__polygon/1`

- `geospatial_protocol`

4.800 `is__void/1`

- `java_utils_protocol`

4.801 `is__vowel/1`

- `characterp`

4.802 `is__white__space/1`

- `characterp`

4.803 `iterator__element/2`

- `java_utils_protocol`

4.804 `jaccard__index/3`

- `string_distance(Representation)`

4.805 `jaro/3`

- `string_distance(Representation)`

4.806 jaro_winkler/3

- string_distance(Representation)

4.807 join/3

- queuep

4.808 join_all/3

- queuep

4.809 jump/3

- queuep

4.810 jump_all/3

- queuep

4.811 jump_all_block/3

- queuep

4.812 k_distinct_subsequence/3

- subsequences_protocol

4.813 k_distinct_subsequences/3

- subsequences_protocol

4.814 k_permutation/3

- permutations_protocol

4.815 k_permutation/4

- permutations_protocol

4.816 k_permutations/3

- permutations_protocol

4.817 k_permutations/4

- permutations_protocol

4.818 key/2

- pairs

4.819 keys/2

- dictionaryp
- pairs

4.820 keys/3

- redis

4.821 keys_values/3

- pairs

4.822 keysort/2

- listp

4.823 kill/1

- process

4.824 kill/2

- process

4.825 known_zone_id/1

- tzif_zone_ids

4.826 kurtosis/2

- statisticsp

4.827 language__object/2

- graph_language_registry

4.828 last/2

- listp
- varlistp

4.829 leaf/1

- hierarchyp

4.830 leaf_class/1

- class_hierarchy

4.831 leaf_classes/1

- class_hierarchy

4.832 leaf_instance/1

- class_hierarchy

4.833 leaf_instances/1

- class_hierarchy

4.834 leap_effective_date/2

- time_scales_data

4.835 leap_offset_at_utc_unix/2

- time_scales_data

4.836 leap_second_date/2

- time_scales_protocol

4.837 leap_seconds_entries/1

- time_scales_data
- time_scales_protocol

4.838 leap_seconds_source/1

- time_scales_data
- time_scales_protocol

4.839 leap_year/1

- datep
- iso8601

4.840 learn/0

- metagol_example_protocol

4.841 learn/1

- metagol_example_protocol

4.842 learn/2

- classifier_protocol
- metagol

4.843 learn/3

- ada_boost
- isolation_forest
- metagol
- random_forest

4.844 learn_seq/2

- metagol

4.845 learn_with_timeout/4

- metagol

4.846 leash/1

- debuggerp

4.847 leashing/1

- debuggerp

4.848 least_common_multiple/2

- numberlistp

4.849 leaves/1

- hierarchyp

4.850 leaves/2

- unweighted_directed_graph(Dictionary)

4.851 length/2

- deque_protocol
- listp
- queuep
- varlistp

4.852 levenshtein/3

- `string__distance(Representation)`

4.853 libraries/0

- `help`

4.854 libraries/1

- `diagram(Format)`
- `diagrams(Format)`
- `lgtdocp`

4.855 libraries/2

- `diagram(Format)`
- `diagrams(Format)`
- `lgtdocp`

4.856 libraries/3

- `diagram(Format)`
- `diagrams(Format)`

4.857 library/1

- `code__metric`
- `dead__code__scanner`
- `diagram(Format)`
- `diagrams(Format)`
- `help`
- `lgtdocp`
- `mutation__testing`

4.858 library/2

- `code_metric`
- `dead_code_scanner`
- `diagram(Format)`
- `diagrams(Format)`
- `lgtdocp`
- `mutation_testing`

4.859 library__mutants/2

- `mutation_testing`

4.860 library__mutants/3

- `mutation_testing`

4.861 library__score/2

- `code_metric`

4.862 license/1

- `application_protocol`
- `pack_protocol`

4.863 linda/0

- `linda_server`

4.864 linda/1

- `linda_server`

4.865 linda_client/1

- linda_client

4.866 linda_client/2

- linda_client

4.867 linda_timeout/2

- linda_client

4.868 line_to_chars/2

- reader

4.869 line_to_chars/3

- reader

4.870 line_to_codes/2

- reader

4.871 line_to_codes/3

- reader

4.872 linear_regression/4

- numberlistp

4.873 lint/0

- packs
- registries

4.874 lint/1

- packs
- registries

4.875 lint/2

- packs

4.876 list/0

- registries

4.877 list__to__array/2

- java_utils_protocol

4.878 listing/0

- listing

4.879 listing/1

- listing

4.880 little_endian_word32/2

- hash_common_32

4.881 `llen/3`

- `redis`

4.882 `load/1`

- `mime_types`
- `tzif_protocol`

4.883 `load/2`

- `mime_types`
- `tzif_protocol`

4.884 `load_config/1`

- `subprocess_mutation_hook`

4.885 `load_coverage_config/1`

- `subprocess_coverage_hook`

4.886 `load_dut1_override/1`

- `time_scales_data`
- `time_scales_protocol`

4.887 `load_leap_seconds_override/1`

- `time_scales_data`
- `time_scales_protocol`

4.888 load_program/1

- datalog_protocol

4.889 loaded_file/1

- logtalk

4.890 loaded_file_property/2

- logtalk
- modules_diagram_support

4.891 loaded_files_topological_sort/1

- logtalk

4.892 loaded_files_topological_sort/2

- logtalk

4.893 loaded_pack/3

- packs

4.894 loaded_pack_dependency/6

- packs

4.895 loaded_pack_file/4

- packs

4.896 loader_file/1

- application_protocol

4.897 local_abbreviation/2

- tzif_protocol

4.898 local_abbreviation/3

- tzif_protocol

4.899 local_abbreviation/4

- tzif_protocol

4.900 local_abbreviation_reified/2

- tzif_protocol

4.901 local_abbreviation_reified/3

- tzif_protocol

4.902 local_abbreviation_reified/4

- tzif_protocol

4.903 local_abbreviation_with_resolution/3

- tzif_protocol

4.904 local_abbreviation_with_resolution/4

- tzif_protocol

4.905 local_abbreviation_with_resolution/5

- tzif_protocol

4.906 local_daylight_saving_time/2

- tzif_protocol

4.907 local_daylight_saving_time/3

- tzif_protocol

4.908 local_daylight_saving_time/4

- tzif_protocol

4.909 local_daylight_saving_time_reified/2

- tzif_protocol

4.910 local_daylight_saving_time_reified/3

- tzif_protocol

4.911 local_daylight_saving_time_reified/4

- tzif_protocol

4.912 local_daylight_saving_time_with_resolution/3

- tzif_protocol

4.913 local_daylight_saving_time_with_resolution/4

- tzif_protocol

4.914 local_daylight_saving_time_with_resolution/5

- tzif_protocol

4.915 local_offset/2

- tzif_protocol

4.916 local_offset/3

- tzif_protocol

4.917 local_offset/4

- tzif_protocol

4.918 local_offset_reified/2

- tzif_protocol

4.919 local_offset_reified/3

- tzif_protocol

4.920 local_offset_reified/4

- tzif_protocol

4.921 local_offset_with_resolution/3

- tzif_protocol

4.922 local_offset_with_resolution/4

- tzif_protocol

4.923 local_offset_with_resolution/5

- tzif_protocol

4.924 local_time_type/2

- tzif_protocol

4.925 local_time_type/3

- tzif_protocol

4.926 local_time_type/4

- tzif_protocol

4.927 local_time_type_reified/2

- tzif_protocol

4.928 local_time_type_reified/3

- tzif_protocol

4.929 local_time_type_reified/4

- tzif_protocol

4.930 local_time_type_with_resolution/3

- tzif_protocol

4.931 local_time_type_with_resolution/4

- tzif_protocol

4.932 local_time_type_with_resolution/5

- tzif_protocol

4.933 local_to_utc/3

- datep

4.934 local_to_utc_tz/3

- dates_tz_protocol

4.935 local_to_utc_tz_with_resolution/4

- dates_tz_protocol

4.936 log/3

- debuggerp

4.937 log_event/2

- loggingp

4.938 log_file/2

- loggingp

4.939 logging/1

- loggingp

4.940 logging/3

- debuggerp

4.941 logistic/3

- sampling_protocol

4.942 lognormal/3

- sampling_protocol

4.943 logseries/2

- sampling_protocol

4.944 logtalk_packs/0

- packs_common

4.945 logtalk_packs/1

- packs_common

4.946 long_flags/1

- command_line_option

4.947 longest_common_increasing_subsequence/3

- subsequences_protocol

4.948 longest_common_subsequence/3

- string_distance(Representation)
- subsequences_protocol

4.949 longest_common_subsequence_length/3

- string_distance(Representation)

4.950 longest_common_substring/3

- string_distance(Representation)

4.951 longest_decreasing_subsequence/2

- subsequences_protocol

4.952 longest_increasing_subsequence/2

- subsequences_protocol

4.953 longest_repeating_subsequence/2

- subsequences_protocol

4.954 lookup/2

- dictionaryp

4.955 lookup/3

- dictionaryp

4.956 lookup/4

- dictionaryp

4.957 lookup_in/3

- nested_dictionary_protocol

4.958 lower_upper/2

- characterp

4.959 lpop/3

- redis

4.960 lpush/4

- redis

4.961 lrange/5

- redis

4.962 lrem/5

- redis

4.963 ltrim/5

- redis

4.964 magic/2

- magic

4.965 magicise/4

- magic

4.966 make_directory/1

- osp

4.967 make_directory_path/1

- osp

4.968 make_set/3

- union_find_protocol

4.969 man/1

- help

4.970 manhattan_distance/3

- numberlistp

4.971 manhattan_norm/2

- numberlistp

4.972 map/2

- deque_protocol
- dictionaryp
- expected(Expected)
- metap
- optional(Optional)
- queuep
- validation(Validation)

4.973 map/3

- deque_protocol
- dictionaryp
- metap
- pairs
- queuep
- validated

4.974 map/4

- metap
- validated

4.975 map/5

- metap

4.976 map/6

- metap

4.977 map/7

- metap

4.978 map/8

- metap

4.979 map_both/3

- expected(Expected)
- validation(Validation)

4.980 map_catching/2

- expected(Expected)
- validation(Validation)

4.981 map_element/2

- java_utils_protocol

4.982 map_invalid/2

- validation(Validation)

4.983 map_or_else/3

- expected(Expected)
- optional(Optional)
- validation(Validation)

4.984 map_reduce/5

- metap

4.985 map_unexpected/2

- expected(Expected)

4.986 mask32/1

- hash_common_32

4.987 mask64/1

- hash_common_64

4.988 materialize/0

- datalog_protocol

4.989 max/2

- listp
- numberlistp
- statisticsp

4.990 max/3

- dictionaryp

4.991 max_clauses/1

- metagol

4.992 max_inv_preds/1

- metagol

4.993 max_path/5

- graph_protocol

4.994 max_size/1

- arbitrary

4.995 max_tree/3

- weighted_undirected_graph(Dictionary)

4.996 maximal_cliques/2

- undirected_graph_common

4.997 maximum_cliques/2

- undirected_graph_common

4.998 maybe/0

- random_protocol

4.999 maybe/1

- random_protocol

4.1000 maybe/2

- random_protocol

4.1001 maybe_call/1

- random_protocol

4.1002 maybe_call/2

- random_protocol

4.1003 mean_center/2

- geospatial_protocol

4.1004 mean_deviation/2

- statisticsp

4.1005 mean_squared_error/3

- statisticsp

4.1006 median/2

- numberlistp
- statisticsp

4.1007 median_deviation/2

- statisticsp

4.1008 meets/2

- intervalp

4.1009 member/2

- listp
- random_protocol
- setp

4.1010 memberchk/2

- listp
- setp
- varlistp

4.1011 merge/3

- heap_protocol

4.1012 message_body/2

- amqp

4.1013 message_delivery_tag/2

- amqp

4.1014 message_exchange/2

- amqp

4.1015 message_hook/4

- logtalk

4.1016 message_prefix_file/6

- logtalk

4.1017 message_prefix_stream/4

- logtalk

4.1018 message_properties/2

- amqp

4.1019 message_property/3

- amqp

4.1020 message_routing_key/2

- amqp

4.1021 message_tokens//2

- logtalk

4.1022 met_by/2

- intervalp

4.1023 meta/1

- command_line_option

4.1024 meta_type/3

- type

4.1025 metaphone/2

- string_distance(Representation)

4.1026 metaphone_match/2

- string_distance(Representation)

4.1027 metarule/6

- metagol

4.1028 metarule_next_id/1

- metagol

4.1029 method/2

- json_rpc

4.1030 method_not_found/2

- json_rpc

4.1031 mget/3

- memcached
- redis

4.1032 mibenum/1

- character_set_protocol

4.1033 midpoint/3

- geospatial_protocol

4.1034 min/2

- listp
- numberlistp
- statisticsp

4.1035 min/3

- dictionaryp

4.1036 min_clauses/1

- metagol

4.1037 min_distances/3

- graph_protocol

4.1038 min_max/3

- numberlistp
- statisticsp

4.1039 min_max_normalization/2

- statisticsp

4.1040 min_path/5

- graph_protocol

4.1041 min_path_bellman_ford/5

- weighted_graph_protocol

4.1042 min_paths/3

- weighted_directed_graph(Dictionary)

4.1043 min_predecessors/3

- graph_protocol

4.1044 min_tree/3

- weighted_undirected_graph(Dictionary)

4.1045 minimum_enclosing_circle/3

- geospatial_protocol

4.1046 modes/2

- numberlistp
- statisticsp

4.1047 module_property/2

- modules_diagram_support

4.1048 monitor/1

- event_registry

4.1049 monitor/4

- event_registry

4.1050 monitor_activated/0

- monitorp

4.1051 monitored/1

- event_registry

4.1052 monitors/1

- event_registry

4.1053 month__weekday__date/5

- datep

4.1054 mset/3

- redis

4.1055 msort/2

- listp

4.1056 msort/3

- listp

4.1057 mul32/3

- hash_common_32

4.1058 mul64/3

- hash_common_64

4.1059 mutation/2

- mutator_protocol

4.1060 mutation/3

- mutations
- mutations_store

4.1061 nack/3

- stomp

4.1062 name/1

- application_protocol
- character_set_protocol
- command_line_option
- pack_protocol
- registry_protocol

4.1063 name_of_day/3

- datep

4.1064 name_of_month/3

- datep

4.1065 natural//1

- number_grammars(Format)

4.1066 nearest_coordinate/5

- geospatial_protocol

4.1067 nearest_point_on_polyline/4

- geospatial_protocol

4.1068 nearest_point_on_segment/4

- geospatial_protocol

4.1069 neighbor_state/2

- simulated_annealing_protocol

4.1070 neighbor_state/3

- simulated_annealing_protocol

4.1071 neighbors/3

- graph_protocol

4.1072 new/1

- graph_protocol
- java_access_protocol
- nested_dictionary_protocol
- streamvars
- term_p

4.1073 new/2

- graph_protocol
- java_access_protocol
- streamvars
- union_find_protocol

4.1074 new/3

- graph_protocol
- intervalp

4.1075 new_line//0

- blank_grammars(Format)

4.1076 new_lines//0

- blank_grammars(Format)

4.1077 next/2

- zipperp

4.1078 next/3

- zipperp

4.1079 next/4

- dictionaryp

4.1080 next_permutation/2

- permutations_protocol

4.1081 nextto/3

- listp
- varlistp

4.1082 node/7

- graph_language_protocol

4.1083 nodebug/0

- debuggerp

4.1084 nolog/3

- debuggerp

4.1085 nologall/0

- debuggerp

4.1086 non__blank//1

- blank_grammars(Format)

4.1087 non__blanks//1

- blank_grammars(Format)

4.1088 nonempty__subsequence/2

- subsequences_protocol

4.1089 nonempty__subsequences/2

- subsequences_protocol

4.1090 normal/3

- `sampling_protocol`

4.1091 normal_element/2

- `html`

4.1092 normalize/2

- `url(Representation)`

4.1093 normalize_coordinate/2

- `geospatial_protocol`

4.1094 normalize_date_time/2

- `datep`

4.1095 normalize_polygon_orientation/3

- `geospatial_protocol`

4.1096 normalize_range/2

- `numberlistp`

4.1097 normalize_range/4

- `numberlistp`

4.1098 normalize__scalar/2

- numberlistp

4.1099 normalize__unit/2

- numberlistp

4.1100 nospy/1

- debuggerp

4.1101 nospy/3

- debuggerp

4.1102 nospy/4

- debuggerp

4.1103 nospyall/0

- debuggerp

4.1104 not64/2

- hash__common__64

4.1105 note/2

- registry__protocol

4.1106 note/3

- pack_protocol

4.1107 notification/2

- json_rpc

4.1108 notification/3

- json_rpc

4.1109 notrace/0

- debuggerp

4.1110 now/3

- timep

4.1111 nth0/3

- listp
- varlistp

4.1112 nth0/4

- listp
- varlistp

4.1113 nth1/3

- listp
- varlistp

4.1114 nth1/4

- listp
- varlistp

4.1115 nth_combination/4

- combinations_protocol

4.1116 nth_permutation/3

- permutations_protocol

4.1117 null/1

- java_utils_protocol

4.1118 null_device_path/1

- osp

4.1119 number//1

- number_grammars(Format)

4.1120 number_of_edges/2

- graph_protocol

4.1121 number_of_tests/1

- lgtunit

4.1122 number_of_vertices/2

- graph_protocol

4.1123 number_string/2

- string(Representation)

4.1124 numbervars/1

- term

4.1125 numbervars/3

- term

4.1126 occurrences/2

- listp

4.1127 occurrences/3

- listp

4.1128 occurs/2

- term

4.1129 of/2

- optional

4.1130 of__expected/2

- expected

4.1131 of__invalid/2

- validation

4.1132 of__invalids/2

- validation

4.1133 of__unexpected/2

- expected

4.1134 of__valid/2

- validation

4.1135 offset/2

- tzif_protocol

4.1136 offset/3

- time_scales_protocol
- tzif_protocol

4.1137 offset/4

- tzif_protocol

4.1138 one_or_more//0

- sequence_grammars

4.1139 one_or_more//1

- sequence_grammars

4.1140 one_or_more//2

- sequence_grammars

4.1141 operating_system_machine/1

- osp

4.1142 operating_system_name/1

- osp

4.1143 operating_system_release/1

- osp

4.1144 operating_system_type/1

- osp

4.1145 option/2

- options_protocol

4.1146 option/3

- options_protocol

4.1147 or/2

- expected(Expected)
- optional(Optional)
- validation(Validation)

4.1148 or64/3

- hash_common_64

4.1149 or_else/2

- expected(Expected)
- optional(Optional)
- validation(Validation)

4.1150 or_else_call/2

- expected(Expected)
- optional(Optional)
- validation(Validation)

4.1151 or_else_fail/1

- expected(Expected)
- optional(Optional)
- validation(Validation)

4.1152 or__else__get/2

- expected(Expected)
- optional(Optional)
- validation(Validation)

4.1153 or__else__throw/1

- expected(Expected)
- validation(Validation)

4.1154 or__else__throw/2

- expected(Expected)
- optional(Optional)
- validation(Validation)

4.1155 originator/1

- application_protocol

4.1156 orphaned/0

- packs

4.1157 orphaned/2

- packs

4.1158 out/1

- linda_client

4.1159 out/2

- linda_client

4.1160 out_degree/3

- directed_graph_protocol

4.1161 outdated/0

- packs

4.1162 outdated/1

- packs

4.1163 outdated/2

- packs

4.1164 outdated/4

- packs

4.1165 outdated/5

- packs

4.1166 output_file_name/2

- graph_language_protocol

4.1167 output_schema/2

- mcp_tool_protocol

4.1168 overlapped_by/2

- intervalp

4.1169 overlaps/2

- intervalp

4.1170 pack_dependency/6

- packs

4.1171 pack_metadata/4

- packs

4.1172 pack_object/3

- packs

4.1173 pack_property/4

- packs

4.1174 package/1

- application_protocol

4.1175 pad_md/4

- hash_common_32

4.1176 params/2

- json_rpc

4.1177 parent/1

- proto_hierarchy

4.1178 parenthesis/2

- characterp

4.1179 parents/1

- proto_hierarchy

4.1180 parse/2

- avro
- base32
- base58
- base64
- base64url
- base85
- cbor(StringRepresentation)
- ccstds(SecondaryHeaderLength)
- json_ld_protocol
- json_lines_protocol
- json_protocol
- json_schema_protocol
- protobuf
- toml_protocol
- toon_protocol

- `url(Representation)`
- `xml`
- `yaml_protocol`

4.1181 `parse/3`

- `avro`
- `protobuf`
- `xml`

4.1182 `parse/4`

- `command_line_options`

4.1183 `parse/5`

- `command_line_options`

4.1184 `parse_all/2`

- `yaml_protocol`

4.1185 `parse_domain/2`

- `pddl`

4.1186 `parse_domain/3`

- `pddl`

4.1187 `parse_error/1`

- `json_rpc`

4.1188 parse_problem/2

- pddl

4.1189 parse_problem/3

- pddl

4.1190 partial_map/4

- rbtrees

4.1191 partition/3

- either
- validated

4.1192 partition/4

- metap

4.1193 partition/5

- listp

4.1194 partition/6

- metap

4.1195 path/3

- graph_protocol

4.1196 path_concat/3

- osp

4.1197 peek_back/2

- deque_protocol

4.1198 peek_front/2

- deque_protocol

4.1199 percentile/3

- statisticsp

4.1200 permutation/2

- listp
- permutations_protocol
- random_protocol
- varlistp

4.1201 permutation/3

- permutations_protocol

4.1202 permutation_index/3

- permutations_protocol

4.1203 permutations/2

- permutations_protocol

4.1204 permutations/3

- permutations_protocol

4.1205 persist/3

- redis

4.1206 pid/1

- osp

4.1207 pin/0

- packs_common

4.1208 pin/1

- packs_common

4.1209 pinned/1

- packs_common

4.1210 plus/3

- integer

4.1211 point_in_polygon/2

- geospatial_protocol

4.1212 point_to_polyline_distance/3

- geospatial_protocol

4.1213 poisson/2

- sampling_protocol

4.1214 polygon_area/2

- geospatial_protocol

4.1215 polygon_bounding_box/2

- geospatial_protocol

4.1216 polygon_centroid/2

- geospatial_protocol

4.1217 polygon_orientation/2

- geospatial_protocol

4.1218 polygon_perimeter/2

- geospatial_protocol

4.1219 polygon_perimeter/3

- geospatial_protocol

4.1220 polygons_intersect/2

- geospatial_protocol

4.1221 polyline_length/2

- geospatial_protocol

4.1222 polyline_length/3

- geospatial_protocol

4.1223 polyline_resample/3

- geospatial_protocol

4.1224 polyline_simplify/3

- geospatial_protocol

4.1225 polyline_split_at_distance/4

- geospatial_protocol

4.1226 pop_back/3

- deque_protocol

4.1227 pop_front/3

- deque_protocol

4.1228 port/5

- ports_profiler

4.1229 portray_clause/1

- listing

4.1230 postorder/2

- bintree

4.1231 power/2

- sampling_protocol

4.1232 power_sequence/4

- integer

4.1233 power_set/2

- subsequences_protocol

4.1234 powerset/2

- setp

4.1235 pp/1

- xml

4.1236 pprint/1

- metagol

4.1237 predicate/1

- caller_diagram(Format)

4.1238 predicate/2

- caller_diagram(Format)
- dead_code_scanner
- mutation_testing

4.1239 predicate/3

- dead_code_scanner
- mutation_testing

4.1240 predicate_info_pair_score_hook/4

- doc_metric

4.1241 predicate_info_score_hook/3

- doc_metric

4.1242 predicate__mode__score__hook/3

- doc__metric

4.1243 predicate__mode__score__hook/5

- doc__metric

4.1244 predicate__mutants/3

- mutation__testing

4.1245 predicate__mutants/4

- mutation__testing

4.1246 predicate__stratum/3

- datalog__protocol

4.1247 predicates/2

- dead__code__scanner

4.1248 predicates/3

- dead__code__scanner

4.1249 predict/3

- classifier__protocol

4.1250 predict/4

- isolation_forest
- knn
- nearest_centroid

4.1251 predict_probabilities/3

- ada_boost
- knn
- naive_bayes
- nearest_centroid
- random_forest

4.1252 predict_probabilities/4

- knn
- nearest_centroid

4.1253 preferred_mime_name/1

- character_set_protocol

4.1254 prefix/0

- packs_common

4.1255 prefix/1

- packs_common

4.1256 prefix/2

- listp
- varlistp

4.1257 prefix/3

- listp

4.1258 preorder/2

- bintree

4.1259 prepend/3

- memcached

4.1260 previous/2

- zipperp

4.1261 previous/3

- zipperp

4.1262 previous/4

- dictionaryp

4.1263 previous_permutation/2

- permutations_protocol

4.1264 print_classifier/1

- classifier_protocol

4.1265 print_flags/0

- flags
- flags_validator

4.1266 print_flags/1

- flags

4.1267 print_message/3

- logtalk

4.1268 print_message_token/4

- logtalk

4.1269 print_message_tokens/3

- logtalk

4.1270 product/2

- numberlistp
- statisticsp

4.1271 product/3

- setp

4.1272 `program_to_clauses/2`

- `metagol`

4.1273 `progress/5`

- `simulated_annealing_protocol`

4.1274 `prompt_get/3`

- `mcp_prompt_protocol`

4.1275 `prompts/1`

- `mcp_prompt_protocol`

4.1276 `proper_prefix/2`

- `listp`

4.1277 `proper_prefix/3`

- `listp`

4.1278 `proper_subsequence/2`

- `subsequences_protocol`

4.1279 `proper_suffix/2`

- `listp`

4.1280 proper_suffix/3

- listp

4.1281 property/1

- application_protocol

4.1282 prove/2

- interpreterp

4.1283 prove/3

- interpreterp

4.1284 provides/2

- registries

4.1285 prune/3

- c45

4.1286 prune/5

- c45

4.1287 push_back/3

- deque_protocol

4.1288 push_front/3

- deque_protocol

4.1289 quartiles/4

- statisticsp

4.1290 query/1

- datalog_protocol

4.1291 query/2

- datalog_protocol

4.1292 question_hook/6

- logtalk

4.1293 question_prompt_stream/4

- logtalk

4.1294 queue_bind/4

- amqp

4.1295 queue_declare/3

- amqp

4.1296 queue_delete/3

- amqp

4.1297 queue_purge/2

- amqp

4.1298 queue_unbind/4

- amqp

4.1299 quick_check/1

- lgtunit

4.1300 quick_check/2

- lgtunit

4.1301 quick_check/3

- lgtunit

4.1302 random/1

- random_protocol

4.1303 random/3

- random_protocol

4.1304 random_combination/3

- combinations_protocol

4.1305 random_node/1

- uuid_protocol

4.1306 random_permutation/2

- permutations_protocol

4.1307 random_subsequence/2

- subsequences_protocol

4.1308 random_tree/1

- benchmark_generators

4.1309 randomize/1

- fast_random(Algorithm)
- random(Algorithm)

4.1310 randseq/4

- random_protocol

4.1311 randset/4

- random_protocol

4.1312 range/2

- statisticsp

4.1313 rank__correlation/3

- statisticsp

4.1314 rd/1

- linda_client

4.1315 rd/2

- linda_client

4.1316 rd_list/2

- linda_client

4.1317 rd_list/3

- linda_client

4.1318 rd_noblock/1

- linda_client

4.1319 rd_noblock/2

- linda_client

4.1320 rdirectories/1

- lgtdocp

4.1321 rdirectories/2

- lgtdocp

4.1322 rdirectory/1

- code__metric
- dead__code__scanner
- diagram(Format)
- diagrams(Format)
- lgtdocp
- wrapper

4.1323 rdirectory/2

- code__metric
- dead__code__scanner
- diagram(Format)
- diagrams(Format)
- lgtdocp
- wrapper

4.1324 rdirectory/3

- diagram(Format)
- diagrams(Format)

4.1325 rdirectory_score/2

- code_metric

4.1326 reachable/3

- graph_protocol

4.1327 read_file/2

- csv_protocol
- read_file
- tsv_protocol

4.1328 read_file/3

- csv_protocol
- tsv_protocol

4.1329 read_file_by_line/2

- csv_protocol
- tsv_protocol

4.1330 read_file_by_line/3

- csv_protocol
- tsv_protocol

4.1331 read_framed_message/2

- json_rpc

4.1332 read_from_atom/2

- term_io_protocol

4.1333 read_from_chars/2

- term_io_protocol

4.1334 read_from_codes/2

- term_io_protocol

4.1335 read_message/2

- json_rpc

4.1336 read_mime_types/2

- mime_types

4.1337 read_only_device_path/1

- osp

4.1338 read_stream/2

- csv_protocol
- tsv_protocol

4.1339 read_stream/3

- csv_protocol
- tsv_protocol

4.1340 read_stream_by_line/2

- csv_protocol
- tsv_protocol

4.1341 read_stream_by_line/3

- csv_protocol
- tsv_protocol

4.1342 read_term_from_atom/3

- term_io_protocol

4.1343 read_term_from_chars/3

- term_io_protocol

4.1344 read_term_from_chars/4

- term_io_protocol

4.1345 read_term_from_codes/3

- term_io_protocol

4.1346 read_term_from_codes/4

- term_io_protocol

4.1347 readme/1

- packs_common

4.1348 readme/2

- packs_common

4.1349 receive/3

- amqp
- stomp

4.1350 recorda/2

- recorded_database_core

4.1351 recorda/3

- recorded_database_core

4.1352 recorded/2

- recorded_database_core

4.1353 recorded/3

- recorded_database_core

4.1354 recordz/2

- recorded_database_core

4.1355 recordz/3

- recorded_database_core

4.1356 relative_standard_deviation/2

- statisticsp

4.1357 release/1

- amqp_pool

4.1358 release_date/1

- application_protocol

4.1359 removeDependent/1

- subject

4.1360 remove_duplicates/2

- listp
- varlistp

4.1361 remove_rule/1

- datalog_protocol

4.1362 rename/4

- redis

4.1363 rename_file/2

- osp

4.1364 replace/3

- zipperp

4.1365 replace/5

- memcached

4.1366 replace_sub_atom/4

- atom

4.1367 report_directory/3

- mutation_testing

4.1368 report_entity/3

- mutation_testing

4.1369 report_library/3

- mutation_testing

4.1370 report_predicate/4

- mutation_testing

4.1371 repository/1

- application_protocol

4.1372 repository_branch/1

- application_protocol

4.1373 repository_commit/1

- application_protocol

4.1374 repository_commit_abbreviated/1

- application_protocol

4.1375 repository_commit_author/1

- application_protocol

4.1376 repository_commit_date/1

- application_protocol

4.1377 repository_commit_message/1

- application_protocol

4.1378 request/3

- json_rpc

4.1379 request/4

- json_rpc

4.1380 rescale/3

- numberlistp

4.1381 reset/0

- counter
- debuggerp
- linter_reporter
- mime_types
- mutator_protocol
- packs_common
- ports_profiler

4.1382 reset/1

- ports_profiler

4.1383 reset_counter/1

- counters

4.1384 reset_counters/0

- counters

4.1385 reset_flags/0

- flags

4.1386 reset_flags/1

- flags

4.1387 reset_genint/0

- genint_core

4.1388 reset_genint/1

- genint_core

4.1389 reset_gensym/0

- gensym_core

4.1390 reset_gensym/1

- gensym_core

4.1391 reset_monitor/0

- monitorp

4.1392 reset_seed/0

- fast_random(Algorithm)
- random(Algorithm)

4.1393 resize/2

- amqp_pool

4.1394 resource_read/3

- mcp_resource_protocol

4.1395 resources/1

- `mcp_resource_protocol`

4.1396 response/3

- `json_rpc`

4.1397 restore/1

- `packs`

4.1398 restore/2

- `packs`

4.1399 result/2

- `json_rpc`

4.1400 retract_fact/1

- `datalog_protocol`

4.1401 reverse/2

- `listp`
- `varlistp`

4.1402 rewind/2

- `zipperp`

4.1403 rewind/3

- zipperp

4.1404 rhumb_bearing/3

- geospatial_protocol

4.1405 rhumb_destination_point/4

- geospatial_protocol

4.1406 rhumb_distance/3

- geospatial_protocol

4.1407 rhumb_midpoint/3

- geospatial_protocol

4.1408 rlibraries/1

- lgtdocp

4.1409 rlibraries/2

- lgtdocp

4.1410 rlibrary/1

- code_metric
- dead_code_scanner
- diagram(Format)
- diagrams(Format)
- lgtdocp

4.1411 rlibrary/2

- code_metric
- dead_code_scanner
- diagram(Format)
- diagrams(Format)
- lgtdocp

4.1412 rlibrary_score/2

- code_metric

4.1413 rol32/3

- hash_common_32

4.1414 rol64/3

- hash_common_64

4.1415 rollback/0

- datalog_protocol

4.1416 root_mean_squared_error/3

- statisticsp

4.1417 ror32/3

- hash_common_32

4.1418 route_distance/2

- geospatial_protocol

4.1419 route_distance/3

- geospatial_protocol

4.1420 route_distance/4

- geospatial_protocol

4.1421 rpop/3

- redis

4.1422 rpush/4

- redis

4.1423 rule/2

- databasep

4.1424 rule/3

- databasep

4.1425 rule/4

- databasep

4.1426 rules/1

- `datalog_protocol`

4.1427 run/0

- `lgtunit`

4.1428 run/1

- `lgtunit`

4.1429 run/2

- `lgtunit`
- `simulated_annealing(Problem,RandomAlgorithm)`

4.1430 run/3

- `simulated_annealing(Problem,RandomAlgorithm)`

4.1431 run/4

- `simulated_annealing(Problem,RandomAlgorithm)`

4.1432 run_test_sets/1

- `lgtunit`

4.1433 sadd/4

- `redis`

4.1434 same_instant/2

- datep

4.1435 same_length/2

- listp
- varlistp

4.1436 same_length/3

- listp

4.1437 save/0

- wrapper

4.1438 save/1

- packs
- tzif_protocol
- wrapper

4.1439 save/2

- packs
- tzif_protocol

4.1440 save_dut1_entries/1

- time_scales_data
- time_scales_protocol

4.1441 save_leap_seconds_entries/1

- time_scales_data
- time_scales_protocol

4.1442 scalar_product/3

- numberlistp

4.1443 scan_left/4

- metap

4.1444 scan_left_1/3

- metap

4.1445 scan_right/4

- metap

4.1446 scan_right_1/3

- metap

4.1447 scard/3

- redis

4.1448 score/3

- isolation_forest

4.1449 score_all/3

- isolation_forest

4.1450 search/1

- packs

4.1451 secondary_header/2

- cclds(SecondaryHeaderLength)

4.1452 secondary_header_flag/2

- cclds(SecondaryHeaderLength)

4.1453 secondary_header_time/2

- cclds(SecondaryHeaderLength)

4.1454 select/3

- listp
- random_protocol
- setp
- varlistp

4.1455 select/4

- listp
- random_protocol

4.1456 selectchk/3

- listp
- setp

4.1457 selectchk/4

- listp

4.1458 send/3

- redis

4.1459 send/4

- stomp

4.1460 send_heartbeat/1

- amqp
- stomp

4.1461 sequence/2

- either
- maybe
- validated

4.1462 sequence/3

- integer

4.1463 sequence/4

- float
- integer
- random_protocol

4.1464 sequence/5

- float

4.1465 sequence_count/2

- ccstds(SecondaryHeaderLength)

4.1466 sequence_flags/2

- ccstds(SecondaryHeaderLength)

4.1467 sequential_occurrences/2

- listp

4.1468 sequential_occurrences/3

- listp

4.1469 serve/3

- queuep

4.1470 server_accept/4

- socket

4.1471 server__accept/5

- socket

4.1472 server__close/1

- socket

4.1473 server__open/2

- socket

4.1474 server__open/3

- socket

4.1475 server__open/4

- socket

4.1476 set/1

- counter

4.1477 set/3

- memcached

4.1478 set/4

- random__protocol
- redis

4.1479 set/5

- memcached

4.1480 set__element/2

- java__utils__protocol

4.1481 set__field/2

- java__access__protocol

4.1482 set__flag__value/2

- flags

4.1483 set__flag__value/3

- flags

4.1484 set__monitor/4

- event__registry

4.1485 set__seed/1

- arbitrary
- pseudo__random__protocol

4.1486 set__spy__point/4

- monitorp

4.1487 set_write_max_depth/1

- debuggerp

4.1488 setrange/5

- redis

4.1489 setup/0

- packs_common

4.1490 shell/1

- osp

4.1491 shell/2

- osp

4.1492 shell_command/1

- doclet

4.1493 shl64/3

- hash_common_64

4.1494 short_flags/1

- command_line_option

4.1495 shr64/3

- hash_common_64

4.1496 shrink/3

- arbitrary

4.1497 shrink_sequence/3

- arbitrary

4.1498 shrinker/1

- arbitrary

4.1499 shutdown_server/1

- linda_client

4.1500 sign//1

- number_grammars(Format)

4.1501 singletons/2

- term_p

4.1502 sismember/4

- redis

4.1503 size/2

- dictionaryp
- heap_protocol
- setp

4.1504 skewness/2

- statisticsp

4.1505 sleep/1

- osp

4.1506 sliding_window/3

- subsequences_protocol

4.1507 smembers/3

- redis

4.1508 softmax/2

- numberlistp

4.1509 softmax/3

- numberlistp

4.1510 software_heritage_identifier/1

- application_protocol

4.1511 sort/2

- listp

4.1512 sort/3

- listp

4.1513 sort/4

- listp

4.1514 soundex/2

- string_distance(Representation)

4.1515 soundex_match/2

- string_distance(Representation)

4.1516 source_file_extension/1

- modules_diagram_support

4.1517 space//0

- blank_grammars(Format)

4.1518 spaces//0

- blank_grammars(Format)

4.1519 split/3

- atom

4.1520 split/4

- listp

4.1521 split_string/4

- string(Representation)

4.1522 spy/1

- debuggerp

4.1523 spy/3

- debuggerp

4.1524 spy/4

- debuggerp

4.1525 spy_point/4

- monitorp

4.1526 spying/1

- debuggerp

4.1527 spying/3

- debuggerp

4.1528 spying/4

- debuggerp

4.1529 srem/4

- redis

4.1530 standard_cauchy/3

- sampling_protocol

4.1531 standard_deviation/2

- statisticsp

4.1532 standard_error/2

- statisticsp

4.1533 standard_exponential/1

- sampling_protocol

4.1534 standard_gamma/2

- sampling_protocol

4.1535 standard_normal/1

- sampling_protocol

4.1536 standard_t/2

- sampling_protocol

4.1537 start/0

- ports_profiler
- shell

4.1538 start/2

- mcp_server

4.1539 start/3

- mcp_server

4.1540 start/4

- mcp_server

4.1541 start/5

- mcp_server

4.1542 start_redirect_to_file/2

- dump_trace

4.1543 started_by/2

- intervalp

4.1544 starts/2

- intervalp

4.1545 state_energy/2

- simulated_annealing_protocol

4.1546 stats/1

- amqp_pool

4.1547 stats/2

- memcached

4.1548 stats/3

- memcached

4.1549 stem/2

- stemmer_protocol

4.1550 stems/2

- stemmer_protocol

4.1551 stop/0

- ports_profiler

4.1552 stop_condition/3

- simulated_annealing_protocol

4.1553 stop_redirect_to_file/0

- dump_trace

4.1554 strata/1

- datalog_protocol

4.1555 stream_to_bytes/2

- reader

4.1556 stream_to_bytes/3

- reader

4.1557 stream_to_chars/2

- reader

4.1558 stream_to_chars/3

- reader

4.1559 stream_to_codes/2

- reader

4.1560 stream_to_codes/3

- reader

4.1561 stream_to_terms/2

- reader

4.1562 stream_to_terms/3

- reader

4.1563 string_chars/2

- string(Representation)

4.1564 string_codes/2

- string(Representation)

4.1565 string_concat/3

- string(Representation)

4.1566 string_length/2

- string(Representation)

4.1567 string_lower/2

- string(Representation)

4.1568 string_upper/2

- string(Representation)

4.1569 strlen/3

- redis

4.1570 strongly_connected_components/2

- directed_graph_protocol

4.1571 sub_string/5

- string(Representation)

4.1572 subclass/1

- class_hierarchy

4.1573 subclasses/1

- class_hierarchy

4.1574 sublist/2

- listp
- varlistp

4.1575 subscribe/4

- stomp

4.1576 subsequence/2

- subsequences__protocol

4.1577 subsequence/3

- listp
- subsequences__protocol

4.1578 subsequence/4

- listp

4.1579 subsequence__at__indices/3

- subsequences__protocol

4.1580 subsequence__length/2

- subsequences__protocol

4.1581 subsequences/2

- subsequences__protocol

4.1582 subsequences/3

- subsequences__protocol

4.1583 subsequences_with_min_span/3

- subsequences_protocol

4.1584 subset/2

- setp

4.1585 subslices/2

- subsequences_protocol

4.1586 substitute/4

- listp

4.1587 subsumes/2

- term_p

4.1588 subterm/2

- term_p
- xml

4.1589 subtract/3

- listp
- setp
- varlistp

4.1590 subtract_duration/3

- datep

4.1591 succ/2

- integer

4.1592 suffix/2

- listp
- varlistp

4.1593 suffix/3

- listp

4.1594 suffix_alias/2

- mime_types

4.1595 sum/2

- numberlistp
- statisticsp

4.1596 sum_of_squares/2

- statisticsp

4.1597 summary/1

- linter_reporter

4.1598 superclass/1

- class_hierarchy

4.1599 superclasses/1

- `class_hierarchy`

4.1600 supplier/1

- `application_protocol`

4.1601 supported_range/2

- `time_scales_protocol`

4.1602 suspend_monitor/0

- `monitorp`

4.1603 swap/1

- `expected(Expected)`
- `validation(Validation)`

4.1604 swap/2

- `random_protocol`

4.1605 swap_consecutive/2

- `random_protocol`

4.1606 symdiff/3

- `setp`

4.1607 `symmetric_closure/2`

- `directed_graph_protocol`

4.1608 `tab//0`

- `blank_grammars(Format)`

4.1609 `tabs//0`

- `blank_grammars(Format)`

4.1610 `tai_minus_utc_for_tai_unix/2`

- `time_scales_data`

4.1611 `tail/2`

- `subsequences_protocol`

4.1612 `tail1/2`

- `subsequences_protocol`

4.1613 `tails/2`

- `subsequences_protocol`

4.1614 `tails1/2`

- `subsequences_protocol`

4.1615 take/3

- listp

4.1616 take/4

- listp

4.1617 tcb_minus_tdb_approx/3

- time_scales_data

4.1618 tcg_minus_tt_approx/3

- time_scales_data

4.1619 tdb_minus_tt_approx/3

- time_scales_data

4.1620 temporary_directory/1

- osp

4.1621 term/2

- sarif

4.1622 term/4

- sarif

4.1623 term_expansion/2

- expanding

4.1624 terms_to_array/2

- java_utils_protocol

4.1625 test/1

- lgtunit

4.1626 time_stamp/1

- osp

4.1627 time_string/3

- iso8601

4.1628 time_type/2

- tzif_protocol

4.1629 time_type/3

- tzif_protocol

4.1630 time_type/4

- tzif_protocol

4.1631 timeout/1

- metagol

4.1632 timestamp/2

- ulid_protocol

4.1633 timestamp/8

- ulid_protocol

4.1634 to_expected/1

- validation(Validation)

4.1635 to_expected/2

- optional(Optional)

4.1636 to_optional/1

- expected(Expected)
- validation(Validation)

4.1637 today/3

- datep

4.1638 tolerance_equal/4

- lgtunit
- number

4.1639 tool/1

- help

4.1640 tool_call/3

- mcp_tool_protocol

4.1641 tool_call/4

- mcp_tool_protocol

4.1642 tools/0

- help

4.1643 tools/1

- mcp_tool_protocol

4.1644 top/3

- heap_protocol

4.1645 top_next/5

- heap_protocol

4.1646 topological_sort/2

- directed_graph_protocol

4.1647 topological_sort/3

- unweighted_directed_graph(Dictionary)

4.1648 touch/3

- memcached

4.1649 trace/0

- debuggerp

4.1650 trace_event/2

- logtalk

4.1651 transitive_closure/2

- directed_graph_protocol

4.1652 transitive_reduction/2

- unweighted_directed_graph(Dictionary)

4.1653 transpose/2

- directed_graph_protocol
- pairs

4.1654 traverse/3

- either
- maybe
- validated

4.1655 triangular/4

- sampling_protocol

4.1656 trim/2

- string(Representation)

4.1657 trim/3

- string(Representation)

4.1658 trim_left/2

- string(Representation)

4.1659 trim_left/3

- string(Representation)

4.1660 trim_right/2

- string(Representation)

4.1661 trim_right/3

- string(Representation)

4.1662 trimmed_mean/3

- statisticsp

4.1663 true/1

- java_utils_protocol

4.1664 tt_minus_tai/2

- time_scales_data

4.1665 ttl/3

- redis

4.1666 tx_commit/1

- amqp

4.1667 tx_rollback/1

- amqp

4.1668 tx_select/1

- amqp

4.1669 type/1

- command_line_option
- type

4.1670 type/2

- ccstds(SecondaryHeaderLength)

4.1671 type/3

- redis

4.1672 tzdb_version/1

- tzif_zone_ids

4.1673 unexpected/1

- expected(Expected)

4.1674 unexpecteds/2

- either

4.1675 uniform/1

- sampling_protocol

4.1676 uniform/3

- sampling_protocol

4.1677 uninstall/0

- packs

4.1678 uninstall/1

- packs

4.1679 `uninstall/2`

- `packs`

4.1680 `union/3`

- `setp`
- `two3tree`
- `unweighted_directed_graph(Dictionary)`

4.1681 `union/4`

- `setp`
- `union_find_protocol`

4.1682 `union_all/3`

- `union_find_protocol`

4.1683 `unix_to_date_time/2`

- `datep`

4.1684 `unpin/0`

- `packs_common`

4.1685 `unpin/1`

- `packs_common`

4.1686 unsubscribe/3

- stomp

4.1687 unzip/2

- zipperp

4.1688 update/0

- doclet
- packs
- registries

4.1689 update/1

- observer
- packs
- registries

4.1690 update/2

- packs
- registries

4.1691 update/3

- datalog_protocol
- dictionaryp
- packs

4.1692 update/4

- dictionaryp

4.1693 update/5

- dictionaryp

4.1694 update_in/4

- nested_dictionary_protocol

4.1695 update_in/5

- nested_dictionary_protocol

4.1696 user__data/2

- ccstds(SecondaryHeaderLength)

4.1697 utc_date_time_to_instant/2

- time_scales_protocol

4.1698 utc_to_local/3

- datep

4.1699 utc_to_local_tz/3

- dates_tz_protocol

4.1700 uuid_max/1

- uuid_protocol

4.1701 uuid_nil/1

- uuid_protocol

4.1702 uuid_null/1

- uuid_protocol

4.1703 uuid_v1/2

- uuid_protocol

4.1704 uuid_v3/3

- uuid_protocol

4.1705 uuid_v4/1

- uuid_protocol

4.1706 uuid_v5/3

- uuid_protocol

4.1707 uuid_v7/1

- uuid_protocol

4.1708 valid/0

- `command_line_option`

4.1709 valid/1

- `intervalp`
- `statisticsp`
- `termp`
- `url(Representation)`
- `validation(Validation)`
- `varlistp`

4.1710 valid/2

- `type`

4.1711 valid/3

- `datep`
- `timep`

4.1712 valid_conversion/3

- `time_scales_protocol`

4.1713 valid_coordinate/1

- `geospatial_protocol`

4.1714 valid_date/3

- `iso8601`

4.1715 valid_date_time/1

- datep

4.1716 valid_instant/1

- time_scales_protocol

4.1717 valid_option/1

- options_protocol

4.1718 valid_options/1

- options_protocol

4.1719 valid_scale/1

- time_scales_protocol

4.1720 valid_until_date/1

- application_protocol

4.1721 validate/1

- flags_validator

4.1722 validate/2

- json_schema_protocol

4.1723 validate/3

- json_schema_protocol

4.1724 valids/2

- validated

4.1725 value/1

- counter

4.1726 value/3

- pairs

4.1727 value_reference/2

- java_utils_protocol

4.1728 values/2

- dictionaryp
- pairs

4.1729 variables/2

- temp

4.1730 variance/2

- statisticsp

4.1731 variant/2

- lgtunit
- termpp

4.1732 varnumbers/2

- termpp

4.1733 varnumbers/3

- termpp

4.1734 verify_commands_availability/0

- packs_common

4.1735 version/1

- application_protocol

4.1736 version/2

- ccstds(SecondaryHeaderLength)
- memcached

4.1737 version/6

- pack_protocol

4.1738 versions/3

- packs

4.1739 vertices/2

- graph_protocol

4.1740 vincenty_distance/3

- geospatial_protocol

4.1741 void/1

- java_utils_protocol

4.1742 void_element/1

- html

4.1743 von_mises/3

- sampling_protocol

4.1744 wait/2

- process

4.1745 wald/3

- sampling_protocol

4.1746 wall_time/1

- osp

4.1747 warning/1

- `linter_reporter`

4.1748 warnings/1

- `linter_reporter`

4.1749 weakly_connected_components/2

- `directed_graph_protocol`

4.1750 week_of_year_iso/2

- `datep`

4.1751 weekday/2

- `datep`

4.1752 weibull/3

- `sampling_protocol`

4.1753 weighted_mean/3

- `statisticsp`

4.1754 welcome/0

- `shell`

4.1755 when/2

- coroutining

4.1756 whiledo/2

- loopp

4.1757 white_space//0

- blank_grammars(Format)

4.1758 white_spaces//0

- blank_grammars(Format)

4.1759 with_connection/1

- amqp_pool

4.1760 with_output_to/2

- term_io_protocol

4.1761 within_distance/4

- geospatial_protocol

4.1762 without//2

- sequence_grammars

4.1763 wneighbors/3

- weighted_graph_protocol

4.1764 word32_hex/2

- hash_common_32

4.1765 word64_hex/2

- hash_common_64

4.1766 working_directory/1

- osp

4.1767 write_file/3

- csv_protocol
- tsv_protocol

4.1768 write_framed_message/2

- json_rpc

4.1769 write_max_depth/1

- debuggerp

4.1770 write_message/2

- json_rpc

4.1771 write_stream/3

- csv_protocol
- tsv_protocol

4.1772 write_term_to_atom/3

- term_io_protocol

4.1773 write_term_to_chars/3

- term_io_protocol

4.1774 write_term_to_chars/4

- term_io_protocol

4.1775 write_term_to_codes/3

- term_io_protocol

4.1776 write_term_to_codes/4

- term_io_protocol

4.1777 write_to_atom/2

- term_io_protocol

4.1778 write_to_chars/2

- term_io_protocol

4.1779 write_to_codes/2

- term_io_protocol

4.1780 xor64/3

- hash_common_64

4.1781 z_normalization/2

- statisticsp

4.1782 zadd/5

- redis

4.1783 zcard/3

- redis

4.1784 zero_or_more//0

- sequence_grammars

4.1785 zero_or_more//1

- sequence_grammars

4.1786 zero_or_more//2

- sequence_grammars

4.1787 zip/2

- zipperp

4.1788 zip/3

- expected(Expected)
- optional(Optional)
- validation(Validation)
- zipperp

4.1789 zip_at_index/4

- zlist

4.1790 zone/3

- tzif_protocol

4.1791 zone_id_kind/2

- tzif_zone_ids

4.1792 zones/1

- tzif_protocol

4.1793 zones/2

- tzif_protocol

4.1794 zrange/5

- redis

4.1795 zrank/4

- redis

4.1796 zrem/4

- redis

4.1797 zscore/4

- redis

INDICES AND TABLES

- `genindex`
- `search`

Generated on Thu Apr 9 17:34:02 WEST 2026

Symbols

(/)/2, 721
 (//)/2, 722
 (<)/2, 1618
 (<=)/2, 56, 973
 (:=)/2, 1619
 (=<)/2, 1618
 (=>)/2, 57, 973
 (=\\=)/2, 1619
 (>)/2, 1618
 (>=)/2, 1619
 =~= / 2, 912, 1661

A

a_star_interpreter(W), 1783
 abbreviation/2, 1717
 abbreviation/3, 1717
 abbreviation/4, 1717
 abort_transaction/3, 1477
 absolute_file_name/2, 1190
 acc_info/5, 479
 acc_info/7, 478
 accept_hook_/1, 992
 ack/3, 1475
 acquire/1, 26
 activate_debug_handler/1, 212
 activate_monitor/0, 491
 active_collection_mode_/1, 872
 active_debug_handler/1, 211
 active_debug_handler_/1, 219
 active_diagnostic_/1, 871
 active_preflight_issue_/1, 871
 ada_boost, 1
 add/1, 1266
 add/2, 1265
 add/3, 1264, 1623
 add/5, 1040
 add32/3, 665
 add32/4, 665
 add32/5, 666
 add64/3, 671
 add_directive_/2, 1825

add_directive_/3, 1826
 add_directive_before_entity_/2, 1825
 add_duration/3, 270
 add_edge/4, 625
 add_edge/5, 638
 add_edges/3, 604
 add_library_documentation_url/4, 421
 add_link_options/3, 366
 add_node_zoom_option/4, 367
 add_rule/3, 259
 add_type/2, 1078
 add_type/3, 1078
 add_vertex/3, 603
 add_vertices/3, 603
 addDependent/1, 328
 after/2, 275, 776
 after/3, 221
 after_event_registry, 480
 aggregate_body_predicate/2, 254
 alias/1, 92
 all/0, 141, 292, 884
 all/1, 140, 292, 884
 all_files/0, 356, 380
 all_files/1, 356, 380
 all_libraries/0, 350, 374
 all_libraries/1, 350, 374
 all_pairs_min_paths/2, 609
 all_pairs_min_predecessors/2, 610
 all_score/1, 143
 along_track_distance/4, 552
 alternating_subsequence/2, 1519
 alternating_subsequences/2, 1519
 amqp, 4
 amqp_pool, 24
 ancestor/1, 735
 ancestor/4, 152
 ancestors/1, 735
 and64/3, 673
 apid/2, 82
 apis/0, 719
 apis/1, 719
 append/2, 1636

- append/3, 1041, 1310, 1636, 1696
- append/4, 1372
- application_/1, 1029
- application_capabilities_/1, 1031
- application_common, 29
- application_protocol, 31
- apply/2, 1845
- apply/4, 455
- approximately_equal/2, 910, 1659
- approximately_equal/3, 910, 1660
- arbitrary, 42
- arbitrary/1, 45
- arbitrary/2, 45
- archive/1, 1277
- arithmetic_mean/2, 1453
- arithmetic_mean/5, 1445
- arithmetic_operator_replacement(Entity, Predicate, ClauseIndex, Occurrence, PrintMutation), 1081
- array_list/2, 817
- array_to_list/2, 816
- array_to_terms/2, 816
- array_to_terms/3, 815
- articulation_points/2, 616
- as_curly_bracketed/2, 446, 1154
- as_curly_bracketed_impl/2, 466
- as_curly_bracketed_impl/3, 466
- as_deque/2, 336
- as_dictionary/2, 446
- as_difflist/2, 1631
- as_heap/2, 707
- as_list/2, 336, 446, 707, 1311, 1400, 1623
- as_nested_dictionary/2, 1154
- as_set/2, 1400
- ask_question/5, 209
- assert_fact/1, 260
- assertion/1, 51, 905
- assertion/2, 51, 905
- assertions, 48
- assertions(Mode), 50
- assertions_messages, 52
- assignable/1, 55
- assignable/2, 56
- assignvars, 53
- assignvarsp, 54
- atom, 1602
- atom_string/2, 1491
- atomic, 1604
- atomics_to_string/2, 1495
- atomics_to_string/3, 1495
- attribute_values/2, 131
- automation_report, 885
- auxiliary_predicate_counter_/1, 943
- available/0, 1213
- available/1, 28, 1213
- available/2, 1212
- average/2, 1667
- average_deviation/3, 1455
- avltree, 440
- avro, 58

B

- backend_adapter_hook, 744
- backend_random, 1313
- base32, 61
- base58, 63
- base64, 65
- base64url, 67
- base85, 69
- base_/2, 539
- baseline_coverage_/4, 1104
- basic_cancel/3, 15
- basic_consume/3, 14
- basic_get/3, 15
- basic_nack/3, 16
- basic_publish/4, 13
- basic_qos/2, 17
- basic_recover/2, 18
- basic_reject/3, 17
- bbox_contains/2, 558
- bbox_expand/3, 559
- bbox_from_coordinates/2, 560
- bbox_intersects/2, 559
- bbox_union/3, 559
- before/2, 275, 776
- before/3, 221
- before_event_registry, 481
- begin/0, 259
- begin_transaction/3, 1476
- bench_goal/1, 1796
- benchmark/2, 908
- benchmark/3, 909
- benchmark/4, 909
- benchmark_generators, 1785
- benchmark_reified/3, 908
- bernoulli/2, 1340
- best_first, 1786
- beta/3, 1340
- between/3, 1327, 1627
- between/4, 1625
- bfs_interpreter, 1788
- big_endian_word32/2, 667
- binary_file_assertion/3, 939
- binary_heap(Order), 698
- binary_heap_max, 699
- binary_heap_min, 701
- binary_input_assertion/2, 924
- binary_input_assertion/3, 923

binary_output_assertion/2, 934
 binary_output_assertion/3, 933
 binary_output_contents/1, 935
 binary_output_contents/2, 934
 binomial/3, 1340, 1658
 bintree, 441
 bit//1, 581
 bits//1, 581
 blank//0, 575
 blank_grammars(Format), 571
 blanks//0, 575
 body_goal_negation(Entity,Predicate,ClauseIndex,Occurrence,Print/But), 1083
 body_pred/1, 1068
 body_pred_call/2, 1070
 bounding_box/3, 564
 branch/2, 568
 breadth_first_order/3, 606
 bridges/2, 616
 built_date/1, 36
 built_in_directive/4, 723
 built_in_flag/2, 525
 built_in_method/4, 724
 built_in_non_terminal/4, 725
 built_in_predicate/4, 724
 bup_interpreter, 1789
 bytes_dword/6, 90
 bytes_hex/2, 664
 bytes_to_codes/2, 94
 bytes_word/4, 89

C

c/1, 1793
 c45, 71
 cache/1, 1710
 cache_source/1, 1712
 cached_run_/5, 871
 cached_tzif/1, 1712
 cached_tzif_/2, 1707
 calendar_month/3, 794
 call_with_timeout/2, 1558
 call_with_timeout/3, 1559
 callable, 1605
 caller_diagram, 337
 caller_diagram(Format), 338
 capabilities/1, 1033
 captured_mutated_terms_/5, 1104
 capturing_mutated_terms_/0, 1104
 cartesian_product/3, 1289
 cas/6, 1042
 cat/2, 1160
 cbor, 74
 cbor(StringRepresentation), 75
 cc_metric, 133
 ccsds, 77
 ccsds(SecondaryHeaderLength), 78
 ccsds_types, 85
 cdata_generation//1, 1834
 central_moment/3, 1462
 central_moment_sum/7, 1447
 change_directory/1, 1194
 changed/0, 327
 changed/1, 328
 channel_close/1, 9
 channel_close/3, 9
 channel_print/But, 1083
 character, 1606
 character_data_format/3, 1834
 character_set, 86
 character_set_protocol, 90
 characterp, 1608
 chebyshev_distance/3, 1669
 chebyshev_norm/2, 1668
 check/0, 193
 check/1, 1685, 1704
 check/2, 1692
 check/3, 1692
 check_binary_file/2, 939
 check_binary_input/1, 923
 check_binary_input/2, 923
 check_binary_output/1, 933
 check_binary_output/2, 932
 check_conversion/3, 1552
 check_convert/4, 1554
 check_instant/1, 1551
 check_offset/3, 1556
 check_option/1, 1178
 check_options/1, 1179
 check_text_file/2, 937
 check_text_file/3, 937
 check_text_input/1, 919
 check_text_input/2, 919
 check_text_output/1, 927
 check_text_output/2, 927
 check_text_output/3, 926
 chi_squared/2, 1338
 chr_is/2, 1582
 chr_next_state/1, 1585
 chr_no_spy/1, 1582
 chr_nospy/0, 1582
 chr_notrace/0, 1582
 chr_option/2, 1583
 chr_option_allow_deep_guards/0, 1584
 chr_option_optimization_level/1, 1583
 chr_option_print_trace/0, 1583
 chr_option_show_history/0, 1584
 chr_option_show_id/0, 1584
 chr_option_show_stack/0, 1584

chr_option_show_store/0, 1584
chr_option_trace_interactive/0, 1583
chr_rule_/1, 1585
chr_spy/1, 1582
chr_spy_point/1, 1585
chr_trace/0, 1582
circular_uniform_cartesian/3, 1345
circular_uniform_polar/3, 1345
class/1, 132, 728
class_/5, 888
class_hierarchy, 725
class_hierarchyp, 727
class_values/1, 132
classes/1, 728
classifier_protocol, 127
classifier_to_clauses/4, 129
classifier_to_file/4, 129
clause/5, 1297
clause_/5, 1298
clause_breakpoint_/2, 306
clause_location/6, 1297
clause_location_/6, 1298
clause_mutator_protocol, 1084
clauses_/1, 1087
clauses_reordering(Entity,Predicate,ClauseIndex,OccurrencePattern/3,186),
1085
clean/0, 1238, 1271
clean/1, 1237, 1270
clean/2, 1237
clean_binary_input/0, 924
clean_binary_output/0, 935
clean_directory/1, 940
clean_file/1, 939
clean_text_input/0, 921
clean_text_output/0, 931
cleanup/0, 914
clear/0, 258
clear_cache/0, 1712
clear_dut1_override/0, 1540, 1549
clear_leap_seconds_override/0, 1538, 1547
client_connection_/3, 991
client_connection_alias_/2, 988
client_connection_input_/2, 987
client_connection_output_/2, 987
client_engine_/2, 993
client_open/4, 1435
client_open/5, 1435
client_timeout_/1, 988
clockwise_polygon/2, 557
clone/1, 967, 1276
clone/3, 447
clone/4, 447
clone_impl/3, 463
clone_impl2/4, 463
cloning, 966
close/1, 7
close/2, 1439
close/3, 7
close_client/1, 979
close_polygon/2, 556
closed_input_stream/2, 940
closed_output_stream/2, 941
cobertura_report, 886
code_metric, 135
code_metrics, 148
code_metrics_messages, 149
code_metrics_utilities, 150
codes_to_bytes/2, 93
coefficient_of_variation/2, 1457
cogc_metric, 156
collect/4, 471
collection_options_/1, 1002
combination/3, 186
combination/4, 187
combination_index/4, 190
combination_with_replacement/3, 188
combination_with_replacement/4, 188
combinations, 183
combinations/3, 186
combinations/4, 186
combinations_protocol, 184
combinations_with_replacement/3, 187
combinations_with_replacement/4, 187
command/2, 1255
command_line_arguments/1, 1204
command_line_option, 192
command_line_options, 197
commit/0, 260
commit_author/2, 568
commit_date/2, 569
commit_hash/2, 569
commit_hash_abbreviated/2, 570
commit_log/3, 570
commit_message/2, 570
commit_transaction/3, 1477
common_subsequence/3, 1515
common_subsequences/3, 1514
compact/3, 832
compare_date_time/3, 276
comparingp, 1617
compile_aux_clauses/1, 216
compile_predicate_heads/4, 217
compile_predicate_indicators/3, 217
compiled_pred_call/2, 1070
complement/2, 626
completion/2, 723
completions/2, 723
compose/3, 620

- compound, 1620
 - compute_ranks/3, 1448
 - condition/0, 914
 - conditional_breakpoint_/3, 308
 - confirm_select/1, 20
 - connect/1, 1369
 - connect/2, 1038
 - connect/3, 1038, 1370
 - connect/4, 6, 1472
 - connected_components/2, 630, 644
 - connection_alive/1, 8, 1473
 - console/1, 1371
 - containment_edge_/2, 344
 - contains/2, 779
 - context_breakpoint_/4, 307
 - context_summaries/2, 1567
 - continuation_byte/1, 88
 - control//0, 576
 - control_construct/4, 724
 - controls//0, 577
 - convert/4, 1554
 - convert_zones/4, 284
 - convert_zones_with_resolution/5, 285
 - cooling_schedule/3, 1419
 - coordinates_bounding_box/2, 554
 - copy_file/2, 1199
 - core_messages, 200
 - coroutining, 223
 - correlation/3, 1461
 - cosine_similarity/3, 1486
 - count_combinations/3, 191
 - count_combinations_with_replacement/3, 191
 - count_distinct_subsequences/3, 1515
 - count_frequencies/4, 1448
 - count_less_equal/6, 1449
 - count_permutations/2, 1291
 - count_subsequences/2, 1521
 - counter, 1790
 - counter/2, 969, 1138
 - counter_/2, 536, 971, 1139
 - counterclockwise_polygon/2, 557
 - counters, 968
 - coupling_metric, 157
 - covariance/3, 1460
 - cover/1, 901
 - coverage_clause_mutator/0, 1116
 - coverage_entry_/4, 1123
 - coverage_file_/1, 1122
 - coverage_report, 890
 - covered_/4, 946
 - cpu_time/1, 279, 1202
 - crc32_non_reflected(Polynomial,Initial,FinalXor,AppendLength), 645
 - crc32_reflected(Polynomial), 647
 - crc32b, 648
 - crc32bzip2, 649
 - crc32c, 651
 - crc32mpeg2, 652
 - crc32posix, 653
 - crc32q, 655
 - create/3, 1300
 - create_binary_file/2, 936
 - create_text_file/2, 936
 - create_text_file/3, 936
 - creators/1, 35
 - cross_deviation_sum/6, 1446
 - cross_track_distance/4, 552
 - csv, 226
 - csv(Header,Separator,IgnoreQuotes), 228
 - csv(Header,Separator,IgnoreQuotes,Comments), 229
 - csv_guess_questions, 231
 - csv_protocol, 232
 - ctrf_output, 892
 - ctrf_report, 893
 - cuid2, 243
 - cuid2(Representation,Size,Alphabet), 244
 - cuid2_protocol, 245
 - current/2, 1842
 - current_entity/1, 152
 - current_host/1, 1439
 - current_predicate_clause_index_/2, 1111
 - current_predicate_directive_index_/2, 1112
 - current_prog/1, 1583
 - current_scope_directive_index_/2, 1112
 - current_uses_directive_index_/2, 1113
 - cycle/2, 594, 615
 - cycle_edge_/2, 369
 - cytoscapejs_graph_language, 341
- ## D
- d2_graph_language, 345
 - damerau_levenshtein/3, 1481
 - data/0, 1295
 - data/1, 1295
 - data/2, 1295
 - data_length/2, 83
 - databasep, 1794
 - datalog, 247
 - datalog_protocol, 256
 - dataset_protocol, 130
 - date, 264
 - date/4, 783
 - date/5, 784
 - date/6, 785
 - date/7, 786
 - date_string/3, 787
 - date_time/7, 1202
 - date_time_string/3, 790

date_time_to_unix/2, 269
datep, 266
dates_tz, 281
dates_tz_protocol, 282
day_of_year/2, 272
day_of_year_date/3, 273
daylight_saving_time/2, 1716
daylight_saving_time/3, 1716
daylight_saving_time/4, 1716
days_in_month/3, 268
deactivate_debug_handler/0, 212
dead_code_scanner, 286
dead_code_scanner_messages, 294
debug/0, 313
debug_expansion(Mode), 1796
debug_handler/1, 211
debug_handler/3, 213
debug_messages, 296
debugger, 300
debugger_messages, 310
debuggerp, 311
debugging/0, 314
debugging/1, 314
debugging_/0, 302
decide/1, 519
decide/2, 519
declares_predicate/2, 152
decode/2, 847
decode_exception/2, 818
decode_exception/3, 819
decode_frame/2, 23
decode_url_spaces/2, 1258
decompile_predicate_heads/4, 217
decompile_predicate_indicators/4, 218
decompose_file_name/3, 1191
decompose_file_name/4, 1191
decr/3, 1374
decr/4, 1045
decrby/4, 1375
decrement_counter/1, 969
default/1, 195
default_atom_mutations, 1128
default_compound_mutations, 1130
default_float_mutations, 1131
default_integer_mutations, 1132
default_list_mutations, 1134
default_option/1, 1180, 1821
default_options/1, 1180, 1821
default_workflow_hook, 746
define_flag/1, 526
define_flag/2, 527
define_log_file/2, 1011
defined/4, 1263
defined_flag/6, 525

defined_flag_/6, 527
defines_predicate/2, 153
defines_predicate/3, 153
degree/3, 629, 643
del/3, 1375
del_monitors/0, 487
del_monitors/4, 487
del_spy_points/4, 492
delete/0, 1270
delete/1, 1269
delete/2, 1044, 1268
delete/3, 1401, 1636, 1697
delete/4, 448, 705
delete_all_after/2, 1849
delete_all_after_and_unzip/2, 1849
delete_all_before/2, 1848
delete_all_before_and_unzip/2, 1848
delete_and_next/2, 1847
delete_and_previous/2, 1847
delete_and_unzip/2, 1848
delete_directory/1, 1193
delete_directory_and_contents/1, 1194
delete_directory_contents/1, 1193
delete_edge/4, 625
delete_edge/5, 639
delete_edges/3, 605
delete_file/1, 1200
delete_impl/5, 464
delete_in/4, 1156
delete_matches/3, 1637
delete_max/4, 453
delete_min/4, 453
delete_vertex/3, 603
delete_vertices/3, 604
demodb, 1797
dependent_/1, 329
dependents/1, 328, 1243
dependents/2, 1243
dependents/3, 1242
depth/2, 1682
depth_first_order/3, 606
deque, 330
deque_protocol, 331
derangement/2, 1290
derangements/2, 1289
derive_aggregate_goals/1, 255
derive_aggregate_literal/2, 254
descendant/1, 736
descendant_class/1, 733
descendant_classes/1, 733
descendant_instance/1, 732
descendant_instances/1, 732
descendants/1, 737
describe/1, 1221, 1262

describe/2, 1220
 description/1, 33, 1207, 1275
 destination_point/4, 551
 destroy/0, 26
 deterministic/1, 904
 deterministic/2, 904
 dfs_interpreter, 1799
 diagnostic/2, 1571
 diagnostic/3, 1570
 diagnostic_rule/5, 1570
 diagnostic_rules/1, 1570
 diagnostic_target/1, 1569
 diagnostics/2, 1572
 diagnostics/3, 1571
 diagnostics_breakdown/2, 1567
 diagnostics_preflight/2, 1573
 diagnostics_preflight/3, 1573
 diagnostics_summary/2, 1572
 diagnostics_summary/3, 1572
 diagnostics_tool/5, 1569
 diagram(*Format*), 346
 diagram_caption/3, 358
 diagram_description/1, 357
 diagram_name_suffix/1, 357
 diagrams, 370
 diagrams(*Format*), 371
 dictionary, 444
 dif, 472
 dif/1, 224, 473
 dif/2, 224, 473
 difflist, 1621
 digit//1, 581
 digits//1, 582
 directed_graph_common, 589
 directed_graph_protocol, 590
 directive_mutator_protocol, 1087
 directories/1, 881, 1818
 directories/2, 352, 376, 880, 1818
 directories/3, 352, 376
 directory/1, 138, 289, 354, 378, 882, 1095, 1251, 1818
 directory/2, 138, 289, 354, 378, 881, 1096, 1251, 1817
 directory/3, 354, 378
 directory_dependency_diagram, 381
 directory_dependency_diagram(*Format*), 382
 directory_diagram(*Format*), 384
 directory_entity_/4, 870
 directory_exists/1, 1197
 directory_files/2, 1196
 directory_files/3, 1197
 directory_load_diagram, 388
 directory_load_diagram(*Format*), 389
 directory_mutants/2, 1102
 directory_mutants/3, 1102
 directory_score/2, 143
 dirichlet/2, 1344
 disable/0, 999
 disable/1, 297
 disable/2, 298
 disable_logging/1, 1013
 disconnect/1, 1038, 1370
 disconnect/2, 1473
 disjoint/2, 1401
 disjoint_sets/2, 1747
 distance/4, 541, 549
 distance/5, 541, 549
 distinct_combination/3, 189
 distinct_combination/4, 189
 distinct_combinations/3, 188
 distinct_combinations/4, 189
 distinct_permutation/2, 1287
 distinct_permutation/3, 1287
 distinct_permutations/2, 1286
 distinct_permutations/3, 1287
 distribution/1, 34
 dit_metric, 159
 djb2_32, 656
 djb2_64, 657
 doc_goal/1, 475
 doc_metric, 160
 doclet, 474
 doctype/1, 766
 document/1, 1393
 document/2, 1393
 document_generation//2, 1833
 dot//1, 584
 dot_graph_language, 391
 double_metaphone/3, 1488
 double_metaphone_match/2, 1489
 dowhile/2, 1016
 downcase_text/2, 1751
 drop/3, 1656
 dump_trace, 323
 duration_between/3, 271
 duration_string/2, 791
 during/2, 779
 dut1_entries/1, 1541, 1550
 dut1_offset_at_utc_unix/3, 1542
 dut1_source/1, 1540, 1549
 dword_bytes/6, 89
 dynamic_directive_/3, 1824

E

easter_day/3, 794
 edb_fact_/1, 249
 edcg, 477
 edge/3, 625
 edge/4, 638
 edge/5, 362

edge/6, 411
edge_/5, 369
edge_case/2, 46
edge_counter_/1, 343
edge_index_/1, 428
edges/2, 602
edit_similarity/3, 1483
edit_similarity/4, 1484
either, 495
either/3, 509
elicit_counter_/1, 1031
elicit_request/5, 1029
empty/1, 332, 449, 602, 706, 1153, 1162, 1307, 1402, 1637, 1697
empty_map/1, 1831
enable/0, 999
enable/1, 297, 999
enable/2, 298
enable_logging/1, 1013
enabled/1, 297
enabled/2, 298
enabled_/0, 1001
enabled_/1, 299
enabled_/2, 299
encode/2, 846
encode_frame/2, 23
encoding_suffix/2, 1080
ensure_directory/1, 1198
ensure_file/1, 1200
entity/1, 137, 288, 438, 721, 1096
entity/2, 288, 438, 1096
entity_calls/3, 154
entity_defines_/2, 1299
entity_diagram, 393
entity_diagram(Format), 394
entity_file_/2, 896
entity_info_pair_score_hook/3, 163
entity_info_score_hook/2, 162
entity_kind/2, 154
entity_mutants/2, 1100
entity_mutants/3, 1100
entity_predicate_breakpoint_/4, 307
entity_predicates_weights_hook/2, 162
entity_prefix/2, 216
entity_property/2, 154
entity_score/2, 141
entity_updates/3, 155
enumerate/2, 1330
environment_variable/2, 1201
epsilon/1, 912
equal/2, 780, 1401
equirectangular_inverse/4, 546
equirectangular_projection/4, 545
erase/1, 1365

error/2, 850
error_code/2, 850
error_data/2, 851
error_message/2, 851
error_response/4, 844
error_response/5, 844
essentially_equal/3, 911, 1660
estimate_temperature/1, 1413
estimate_temperature/2, 1413
euclidean_distance/3, 1669
euclidean_norm/2, 1668
event_registry, 483
event_registryp, 484
example/3, 132
exception/4, 1833
exchange_bind/4, 10
exchange_declare/3, 9
exchange_delete/3, 10
exchange_unbind/4, 11
exclude/3, 1052
execution_context/7, 218
exists/3, 1376
expand/2, 832
expand_library_alias_paths, 493
expand_library_path/2, 213
expanding, 201
expected, 498
expected(Expected), 504
expected/1, 508
expecteds/2, 496
expire/4, 1377
explain//1, 1601
explain/2, 262
explicit_tracing_/0, 302
exponential/2, 1339
export/1, 1394
export/2, 1393
extension/1, 740
extension_type/2, 1076
extension_type/3, 1076
extensions/1, 741
external_predicate_/1, 439
external_reference/2, 37

F

f/4, 1787
factorial/2, 1657
facts/1, 263
fail_insertion(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation)
1089
failed_/3, 945
failed_count_/1, 1125
failed_test_reason//1, 948
false/1, 812

[fast_random](#), 1314
[fast_random\(Algorithm\)](#), 1316
[fault/5](#), 1833
[fcube](#), 517
[fcube/0](#), 518
[file/1](#), 137, 289, 396, 883, 1820
[file/2](#), 137, 288, 395, 883, 1820
[file_being_advised_/4](#), 1826
[file_dependency_diagram](#), 398
[file_dependency_diagram\(Format\)](#), 399
[file_diagram\(Format\)](#), 401
[file_exists/1](#), 1198
[file_footer/3](#), 410
[file_header/3](#), 410
[file_line_hit_count_/3](#), 309
[file_load_diagram](#), 405
[file_load_diagram\(Format\)](#), 406
[file_modification_time/2](#), 1198
[file_path/2](#), 915
[file_path_components/2](#), 1750
[file_permission/2](#), 1199
[file_score/2](#), 142
[file_size/2](#), 1199
[file_to_bytes/2](#), 1354
[file_to_bytes/3](#), 1354
[file_to_chars/2](#), 1353
[file_to_chars/3](#), 1353
[file_to_codes/2](#), 1352
[file_to_codes/3](#), 1352
[file_to_terms/2](#), 1353
[file_to_terms/3](#), 1354
[file_type_extension/2](#), 215
[file_url/2](#), 916
[files/1](#), 356, 380, 882, 1819
[files/2](#), 355, 379, 882, 1819
[files/3](#), 355, 379
[filter/2](#), 1169
[filter/3](#), 512, 1774
[filter_external_file_extension/3](#), 366
[filter_file_extension/3](#), 365
[final_bearing/3](#), 550
[find/4](#), 1746
[find/5](#), 1747
[findall_in_noblock/3](#), 986
[findall_in_noblock/4](#), 986
[findall_member/4](#), 1053
[findall_member/5](#), 1053
[findall_rd_noblock/3](#), 986
[findall_rd_noblock/4](#), 985
[finished_by/2](#), 780
[finishes/2](#), 779
[fips202_hash\(A,B,C\)](#), 659
[fired_/3](#), 946
[fisher/3](#), 1338
[fix_node2_left_underflow/6](#), 464
[fix_node2_right_underflow/6](#), 464
[fix_node3_left_underflow/9](#), 465
[fix_node3_middle_underflow/9](#), 465
[fix_node3_right_underflow/9](#), 465
[fix_option/2](#), 1182
[fix_options/2](#), 1182
[flag_group_chk/1](#), 524
[flag_groups/1](#), 524
[flag_value_/2](#), 527
[flags](#), 521
[flags_validator](#), 528
[flaky_/1](#), 946
[flat_map/2](#), 509, 1170, 1775
[flatten/1](#), 516, 1174, 1782
[flatten/2](#), 832, 1637, 1697
[flatten_goals//1](#), 1801
[flatting](#), 1800
[float](#), 1624
[float//1](#), 583
[flush_all/1](#), 1047
[flush_all/2](#), 1047
[fnv1a_32](#), 660
[fnv1a_64](#), 661
[fold_left/4](#), 1054
[fold_left_1/3](#), 1055
[fold_right/4](#), 1056
[fold_right_1/3](#), 1056
[fordownto/3](#), 1019
[fordownto/4](#), 1019
[fordownto/5](#), 1020
[foreach/3](#), 1017
[foreach/4](#), 1017
[format](#), 530
[format/2](#), 531
[format/3](#), 531
[format_date_time/4](#), 272
[format_entity_score//2](#), 144
[format_object/1](#), 357
[format_report/1](#), 1100
[format_report/2](#), 1099
[format_report/3](#), 1099
[format_to_atom/3](#), 1532
[format_to_chars/3](#), 1532
[format_to_chars/4](#), 1533
[format_to_codes/3](#), 1533
[format_to_codes/4](#), 1533
[forto/3](#), 1018
[forto/4](#), 1018
[forto/5](#), 1018
[forward/1](#), 204
[forward/2](#), 1845
[forward/3](#), 1845
[forwarding](#), 203

fractile/3, 1459
frame_body/2, 1479
frame_command/2, 1478
frame_header/3, 1478
frame_headers/2, 1478
freeze/2, 225
frequency_distribution/2, 1463
from_expected/2, 1769
from_generator/2, 502, 1164, 1768
from_generator/3, 502, 1164, 1768
from_generator/4, 501, 1767
from_goal/2, 501, 1163, 1767
from_goal/3, 501, 1163, 1767
from_goal/4, 500, 1766
from_goal_or_throw/2, 1165
from_goal_or_throw/3, 1164
from_optional/3, 503, 1769
frozen/2, 225
full_device_path/1, 1196
func_test/3, 1068
functional/0, 1068

G

gamma/3, 1341
gat/4, 1046
gats/5, 1046
generate/1, 246, 772, 867, 1146, 1427, 1739
generate/2, 62, 64, 66, 68, 70, 76, 80, 765, 825, 831, 839, 1390, 1565, 1579, 1739, 1750, 1838
generate/3, 59, 81, 1304
generate/4, 60, 1304, 1391
generate/8, 1739
generate_all/2, 1839
generated_predicate_/1, 1063
generating_/0, 956
genint, 532
genint/2, 535
genint_core, 533
gensym, 536
gensym/2, 539
gensym_core, 537
geometric/2, 1339
geometric_mean/2, 1453
geospatial, 540
geospatial_protocol, 542
get/1, 1171
get/3, 1042, 1371
get/4, 1043
get_field/2, 807
get_flag_value/2, 522
get_seed/1, 47, 1320
getrange/5, 1372
gets/4, 1043
gets/5, 1044

git, 566
git_object_identifier/1, 40
git_protocol, 567
gnu/0, 518
goal_expansion/2, 202
grammar_rules_hook, 747
graph_coloring/3, 616
graph_common, 596
graph_footer/5, 410
graph_header/5, 410
graph_language_protocol, 408
graph_language_registry, 412
graph_protocol, 599
graph_types, 612
ground/1, 1683
ground_entity_identifier/3, 365
group_by_key/2, 1679
group_consecutive_by_key/2, 1678
group_sorted_by_key/2, 1678
guess_all_extensions/2, 1077
guess_all_extensions/3, 1078
guess_arity/2, 241
guess_extension/2, 1076
guess_extension/3, 1077
guess_file_type/3, 1075
guess_file_type/4, 1075
guess_separator/2, 241
guess_type/3, 1074
guess_type/4, 1075
gumbel/3, 1344

H

halstead_metric, 165
halstead_metric(Stroud), 166
hamming/3, 1482
hamming_distance/3, 1638
handbook/0, 719
handle_root_underflow/2, 463
harmonic_mean/2, 1454
has_aggregate_rules/0, 254
has_cycle/1, 594, 615
has_negative_cycle/1, 640
has_path/3, 608
hash/2, 676
hash_common_32, 663
hash_common_64, 669
hash_protocol, 675
haversine_distance/3, 546
hdel/4, 1379
head/2, 1308
head_arguments_mutation(Entity,Predicate,ClauseIndex,Occurrence) 1090
head_arguments_reordering(Entity,Predicate,ClauseIndex,Occurrence) 1091

head_pred/1, 1068
 heap(Order), 702
 heap_protocol, 703
 heapp, 708
 help, 717
 help/0, 719, 1248
 help/1, 196
 help/2, 199
 help/3, 199
 heuristic_expansion(Mode), 1801
 hex_digit//1, 582
 hex_digits//1, 582
 hexists/4, 1380
 hget/4, 1379
 hgetall/3, 1379
 hierarchyp, 734
 high_surrogate/1, 88
 hkeys/3, 1380
 hlen/3, 1381
 home/1, 1208, 1276
 homepage/1, 34
 hook_pipeline(Pipeline), 741
 hook_set(Set), 743
 hset/5, 1378
 html, 764
 html5, 767
 hvals/3, 1380
 hypergeometric/4, 1339

I

ibk/3, 1068
 id/2, 849
 idb_fact_/1, 249
 iddfs_interpreter(Increment), 1803
 identity_hook, 748
 ids, 769
 ids(Representation,Bytes), 770
 if_empty/1, 1168
 if_expected/1, 506
 if_expected_or_else/2, 507
 if_invalid/1, 1773
 if_present/1, 1168
 if_present_or_else/2, 1169
 if_unexpected/1, 506
 if_valid/1, 1772
 if_valid_or_else/2, 1773
 in/1, 981
 in/2, 981
 in_degree/3, 593
 in_list/2, 982
 in_list/3, 982
 in_noblock/1, 982
 in_noblock/2, 981
 in_nodes_section_/0, 342
 in_use/2, 29
 include/3, 1052
 included_caller_/1, 340
 included_directory_/1, 387
 included_entity_/1, 396
 included_file_/1, 404
 included_library_/2, 422
 included_module_/1, 396
 included_predicate_/1, 439
 incr/3, 1374
 incr/4, 1045
 incrby/4, 1375
 increase/1, 1792
 increment/0, 1791
 increment_counter/1, 969
 inheritance_diagram, 413
 inheritance_diagram(Format), 415
 init/0, 1813
 init/2, 1506
 init1/2, 1508
 init_log_file/2, 1011
 init_tail/2, 1510
 init_tails/2, 1509
 initial_bearing/3, 550
 initial_state/1, 1417
 initial_temperature/1, 1418
 initialize/1, 25
 initialized_/0, 1029
 inits/2, 1505
 inits1/2, 1507
 inorder/2, 443
 insert/3, 1402
 insert/4, 447, 705
 insert_after/3, 1846
 insert_all/3, 705, 1402
 insert_before/3, 1846
 insert_empty/4, 467
 insert_impl/4, 466
 insert_in/4, 1156
 insert_node2_2/4, 467
 insert_node2_4/4, 467
 insert_node3_4/4, 468
 insert_node3_7/4, 468
 install/1, 1231
 install/2, 1231
 install/3, 1230
 install/4, 1228
 installed/0, 1215
 installed/1, 1215
 installed/3, 1214
 installed/4, 1213
 instance/1, 729
 instance/2, 1365
 instances/1, 729

instant_to_utc_date_time/2, 1554
integer, 1626
integer//1, 583
integer_to_big_endian_bytes32/2, 668
integer_to_big_endian_bytes64/2, 674
integer_to_little_endian_bytes32/2, 668
internal_error/2, 846
internal_os_path/2, 1192
interpolate_great_circle/4, 551
interpolate_rhumb/4, 548
interpreterp, 1804
interquartile_range/2, 1460
intersect/2, 1403
intersection/2, 451
intersection/3, 451, 1403
intersection/4, 1403
intersection_impl/4, 468
interval, 772
interval_string/2, 791
intervalp, 774
invalid/1, 1774
invalid_params/2, 846
invalid_request/1, 845
invalids/2, 1761
invocation_number_/1, 304
invoke/1, 808
invoke/2, 808
ip_grammars(Format), 577
ipv4//1, 578
ipv6//1, 579
is_absolute_file_name/1, 1190
is_acyclic/1, 594
is_alpha/1, 1610
is_alphanumeric/1, 1609
is_ascii/1, 1609
is_batch/1, 848
is_bin_digit/1, 1610
is_bipartite/1, 611
is_clockwise_polygon/1, 556
is_complete/1, 610
is_connected/1, 629, 644
is_control/1, 1614
is_dec_digit/1, 1611
is_empty/0, 1167
is_end_of_line/1, 1615
is_error_response/1, 848
is_expected/0, 506
is_false/1, 814
is_hex_digit/1, 1611
is_invalid/0, 1772
is_layout/1, 1613
is_letter/1, 1610
is_lower_case/1, 1612
is_newline/1, 1615

is_node4/1, 469
is_notification/1, 847
is_null/1, 814
is_object/1, 815
is_octal_digit/1, 1611
is_period/1, 1614
is_prefix_of/2, 1516
is_present/0, 1167
is_punctuation/1, 1614
is_quote/1, 1613
is_request/1, 847
is_response/1, 848
is_sparse/1, 611
is_subsequence_of/2, 1513
is_suffix_of/2, 1516
is_tree/1, 615
is_true/1, 813
is_unexpected/0, 506
is_upper_case/1, 1612
is_valid/0, 1772
is_valid_polygon/1, 558
is_validator/1, 528
is_void/1, 814
is_vowel/1, 1612
is_white_space/1, 1613
iso8601, 781
iso_8859_1, 94
iso_8859_10, 95
iso_8859_13, 97
iso_8859_14, 98
iso_8859_15, 99
iso_8859_16, 100
iso_8859_2, 101
iso_8859_3, 103
iso_8859_4, 104
iso_8859_9, 105
isolation_forest, 796
issue_creator, 800
iterator_element/2, 817

J

jaccard_index/3, 1486
jaro/3, 1482
jaro_winkler/3, 1483
java, 801
java(Reference), 803
java(Reference,ReturnValue), 804
java_access_protocol, 805
java_hook, 809
java_utils_protocol, 810
join/3, 1308
join_all/3, 1308
json, 820

[json\(ObjectRepresentation,PairRepresentation,StringRepresentation\), 732](#)
[822](#)
[json\(StringRepresentation\), 821](#)
[json_ld, 825](#)
[json_ld\(ObjectRepresentation,PairRepresentation,StringRepresentation\), 828](#)
[json_ld\(StringRepresentation\), 827](#)
[json_ld_protocol, 829](#)
[json_lines, 833](#)
[json_lines\(ObjectRepresentation,PairRepresentation,StringRepresentation\), 836](#)
[json_lines\(StringRepresentation\), 835](#)
[json_lines_protocol, 837](#)
[json_protocol, 823](#)
[json_rpc, 840](#)
[json_schema, 853](#)
[json_schema\(ObjectRepresentation,PairRepresentation,StringRepresentation\), 856](#)
[json_schema\(StringRepresentation\), 855](#)
[json_schema_protocol, 857](#)
[jump/3, 1309](#)
[jump_all/3, 1309](#)
[jump_all_block/3, 1310](#)
[jump_to_invocation_number_/1, 305](#)

K

[k_distinct_subsequence/3, 1520](#)
[k_distinct_subsequences/3, 1520](#)
[k_permutation/3, 1288](#)
[k_permutation/4, 1289](#)
[k_permutations/3, 1288](#)
[k_permutations/4, 1288](#)
[key/2, 1676](#)
[keys/2, 454, 1676](#)
[keys/3, 1376](#)
[keys_values/3, 1676](#)
[keysort/2, 1638](#)
[kill/1, 1301](#)
[kill/2, 1301](#)
[knn, 860](#)
[known_zone_id/1, 1734](#)
[ksuid, 863](#)
[ksuid\(Representation,Alphabet\), 864](#)
[ksuid_protocol, 866](#)
[kurtosis/2, 1458](#)

L

[language_object/2, 413](#)
[last/2, 1638, 1698](#)
[last_time_sequence_/2, 1423](#)
[lcom_metric, 168](#)
[lcov_report, 895](#)
[leaf/1, 736](#)
[leaf_class/1, 731](#)
[leaf_instance/1, 731](#)
[leaf_instances/1, 731](#)
[leap_effective_date/2, 1542](#)
[leap_effective_date\(StringRepresentation\), 1541](#)
[leap_second_date/2, 1547](#)
[leap_seconds_entries/1, 1539, 1548](#)
[leap_seconds_source/1, 1538, 1548](#)
[leap_year/1, 267, 793](#)
[learn/0, 1072](#)
[learn/1, 1072](#)
[learn/2, 128, 1066](#)
[learn/3, 3, 798, 1065, 1349](#)
[learn_seq/2, 1066](#)
[learn_with_timeout/4, 1066](#)
[leasing/1, 316](#)
[leasing_/1, 304](#)
[least_common_multiple/2, 1672](#)
[leaves/1, 736](#)
[leaves/2, 621](#)
[length/2, 334, 1310, 1639, 1698](#)
[levenshtein/3, 1481](#)
[lgtdoc, 868](#)
[lgtdoc_messages, 872](#)
[lgtdocp, 873](#)
[lgtunit, 897](#)
[lgtunit_messages, 947](#)
[libraries/0, 720](#)
[libraries/1, 349, 373, 878](#)
[libraries/2, 349, 373, 877](#)
[libraries/3, 349, 373](#)
[library/1, 139, 291, 351, 375, 721, 878, 1095](#)
[library/2, 139, 290, 351, 375, 878, 1095](#)
[library_dependency_diagram, 416](#)
[library_dependency_diagram\(Format\), 417](#)
[library_diagram\(Format\), 419](#)
[library_entity_/4, 869](#)
[library_load_diagram, 423](#)
[library_load_diagram\(Format\), 425](#)
[library_mutants/2, 1101](#)
[library_mutants/3, 1102](#)
[library_score/2, 141](#)
[license/1, 34, 1207](#)
[linda, 974](#)
[linda/0, 991](#)
[linda/1, 990](#)
[linda_client, 976](#)
[linda_client/1, 978](#)
[linda_client/2, 978](#)
[linda_server, 989](#)
[linda_timeout/2, 979](#)
[line_to_chars/2, 1358](#)

line_to_chars/3, 1358
line_to_codes/2, 1358
line_to_codes/3, 1359
linear_regression/4, 1674
lines_covered_/1, 888
lines_metric, 169
lines_total_/1, 889
lint/0, 1245, 1272
lint/1, 1245, 1272
lint/2, 1244
linter_reporter, 997
linter_warning_sequence_/1, 944
list, 1630
list(Type), 1632
list/0, 1262
list_to_array/2, 817
listing, 1003
listing/0, 1004
listing/1, 1004
listp, 1633
literal_bucket/2, 256
little_endian_word32/2, 667
llen/3, 1383
load/1, 1079, 1710
load/2, 1079, 1710
load_config/1, 1124
load_coverage_config/1, 1122
load_dut1_override/1, 1539, 1549
load_leap_seconds_override/1, 1538, 1547
load_program/1, 258
load_registry/1, 1256
loaded_file/1, 213
loaded_file_property/2, 214, 430
loaded_files_topological_sort/1, 215
loaded_files_topological_sort/2, 215
loaded_pack/3, 1224
loaded_pack_dependency/6, 1227
loaded_pack_file/4, 1225
loader_file/1, 35
local_abbreviation/2, 1726
local_abbreviation/3, 1725
local_abbreviation/4, 1725
local_abbreviation_reified/2, 1732
local_abbreviation_reified/3, 1731
local_abbreviation_reified/4, 1731
local_abbreviation_with_resolution/3, 1727
local_abbreviation_with_resolution/4, 1727
local_abbreviation_with_resolution/5, 1726
local_daylight_saving_time/2, 1723
local_daylight_saving_time/3, 1723
local_daylight_saving_time/4, 1723
local_daylight_saving_time_reified/2, 1731
local_daylight_saving_time_reified/3, 1730
local_daylight_saving_time_reified/4, 1730
local_daylight_saving_time_with_resolution/3, 1725
local_daylight_saving_time_with_resolution/4, 1724
local_daylight_saving_time_with_resolution/5, 1724
local_offset/2, 1721
local_offset/3, 1721
local_offset/4, 1720
local_offset_reified/2, 1729
local_offset_reified/3, 1729
local_offset_reified/4, 1729
local_offset_with_resolution/3, 1722
local_offset_with_resolution/4, 1722
local_offset_with_resolution/5, 1721
local_time_type/2, 1719
local_time_type/3, 1718
local_time_type/4, 1718
local_time_type_reified/2, 1728
local_time_type_reified/3, 1728
local_time_type_reified/4, 1727
local_time_type_with_resolution/3, 1720
local_time_type_with_resolution/4, 1719
local_time_type_with_resolution/5, 1719
local_to_utc/3, 271
local_to_utc_tz/3, 283
local_to_utc_tz_with_resolution/4, 284
locate_directory/2, 364
locate_file/5, 365
locate_library/2, 364
log/3, 320
log_event/2, 1012
log_file/2, 1011
log_file_/2, 1007, 1009
log_point_/3, 306
logger, 1005
logging, 1008
logging/1, 1012
logging/3, 320
logging_to_file_/2, 1007, 1009
loggingp, 1010
logistic/3, 1341
lognormal/3, 1337
logseries/2, 1338
logtalk, 205
logtalk_packs/0, 1253
logtalk_packs/1, 1253
long_flags/1, 194
longest_common_increasing_subsequence/3, 1512
longest_common_subsequence/3, 1485, 1510
longest_common_subsequence_length/3, 1484
longest_common_substring/3, 1485
longest_decreasing_subsequence/2, 1511
longest_increasing_subsequence/2, 1511

[longest_repeating_subsequence/2](#), 1512
[lookup/2](#), 450
[lookup/3](#), 450
[lookup/4](#), 450
[lookup_in/3](#), 1154
[loop](#), 1014
[loopp](#), 1015
[lovins_stemmer\(Representation\)](#), 1465
[low_surrogate/1](#), 88
[lower_upper/2](#), 1616
[lpop/3](#), 1382
[lpush/4](#), 1381
[lrange/5](#), 1382
[lrem/5](#), 1383
[ltrim/5](#), 1384

M

[magic](#), 1806
[magic/2](#), 1807
[magic_expansion\(Mode\)](#), 1808
[magicise/4](#), 1807
[make/1](#), 915
[make_directory/1](#), 1192
[make_directory_path/1](#), 1193
[make_set/3](#), 1745
[man/1](#), 722
[manhattan_distance/3](#), 1670
[manhattan_norm/2](#), 1669
[map/2](#), 335, 454, 508, 1058, 1169, 1311, 1775
[map/3](#), 335, 455, 1058, 1312, 1679, 1762
[map/4](#), 1058, 1762
[map/5](#), 1059
[map/6](#), 1059
[map/7](#), 1060
[map/8](#), 1060
[map_both/3](#), 513, 1777
[map_catching/2](#), 512, 1776
[map_element/2](#), 818
[map_impl/2](#), 469
[map_impl/3](#), 469
[map_invalid/2](#), 1777
[map_member/3](#), 1831
[map_or_else/3](#), 514, 1174, 1776
[map_reduce/5](#), 1060
[map_store/4](#), 1832
[map_unexpected/2](#), 512
[mapped_single_byte_character_set](#), 106
[mapping/2](#), 107
[mask32/1](#), 665
[mask64/1](#), 671
[materialize/0](#), 261
[max/2](#), 1452, 1639, 1665
[max/3](#), 453
[max_clauses/1](#), 1069

[max_inv_preds/1](#), 1069
[max_path/5](#), 608
[max_size/1](#), 48
[max_tree/3](#), 645
[maxheap](#), 710
[maximal_cliques/2](#), 617
[maximum_cliques/2](#), 617
[maybe](#), 1158
[maybe/0](#), 1332
[maybe/1](#), 1333
[maybe/2](#), 1333
[maybe_call/1](#), 1333
[maybe_call/2](#), 1334
[mcp_prompt_protocol](#), 1021
[mcp_resource_protocol](#), 1023
[mcp_server](#), 1025
[mcp_tool_protocol](#), 1032
[md5](#), 676
[mean_center/2](#), 553
[mean_deviation/2](#), 1456
[mean_squared_error/3](#), 1464
[median/2](#), 1455, 1667
[median_deviation/2](#), 1456
[meets/2](#), 777
[member/2](#), 1328, 1404, 1639
[memberchk/2](#), 1404, 1640, 1698
[memcached](#), 1036
[merge/3](#), 706
[merge_options/2](#), 1182, 1822
[mermaid_graph_language](#), 427
[message_body/2](#), 21
[message_cache_/1](#), 893, 894, 961, 962, 964, 966
[message_delivery_tag/2](#), 22
[message_diagram_description/1](#), 368
[message_exchange/2](#), 22
[message_hook/4](#), 209
[message_prefix_file/6](#), 209
[message_prefix_stream/4](#), 208
[message_properties/2](#), 21
[message_property/3](#), 21
[message_routing_key/2](#), 22
[message_tokens//2](#), 208
[met_by/2](#), 777
[meta](#), 1049
[meta/1](#), 195
[meta_compiler](#), 1061
[meta_type/3](#), 1691
[metagol](#), 1063
[metagol_example_protocol](#), 1071
[metap](#), 1050
[metaphone/2](#), 1488
[metaphone_match/2](#), 1488
[metarule/6](#), 1067
[metarule_next_id/1](#), 1069

method/2, 849
method_/7, 888
method_not_found/2, 845
mget/3, 1044, 1373
mi_metric, 171
mi_metric(Stroud), 172
mibenum/1, 93
midpoint/3, 550
mime_types, 1073
min/2, 1452, 1640, 1665
min/3, 452
min_clauses/1, 1068
min_distances/3, 609
min_max/3, 1452, 1666
min_max_normalization/2, 1463
min_max_normalize/4, 1447
min_path/5, 608
min_path_bellman_ford/5, 640
min_paths/3, 633
min_predecessors/3, 609
min_tree/3, 644
minheap, 711
minimal_output, 948
minimum_enclosing_circle/3, 553
missing_predicate_directive_/3, 1824
modes/2, 1455, 1667
module_predicate_called_/3, 1823
module_property/2, 430
modules_diagram_support, 429
monitor, 488
monitor/1, 485
monitor/4, 486
monitor_activated/0, 491
monitored/1, 486
monitoring, 220
monitord, 489
monitors/1, 485
month_weekday_date/5, 273
mset/3, 1374
msort/2, 1640
msort/3, 1641
mul32/3, 666
mul64/3, 671
multifile_directive_/3, 1825
murmurhash3_x64_128, 678
murmurhash3_x86_128, 679
murmurhash3_x86_32, 680
mutation/2, 1115
mutation/3, 1136, 1138
mutation/4, 1139
mutation_testing, 1093
mutation_testing_messages, 1105
mutations, 1135
mutations_store, 1137

mutator_common, 1106
mutator_protocol, 1114

N

nack/3, 1476
naive_bayes, 1140
name/1, 33, 92, 194, 1207, 1275
name_of_day/3, 268
name_of_month/3, 268
nanoid, 1142
nanoid(Representation,Size,Alphabet), 1143
nanoid_protocol, 1145
natural, 1656
natural//1, 583
navltree, 1149
nbintree, 1150
nearest_centroid, 1146
nearest_coordinate/5, 553
nearest_point_on_polyline/4, 561
nearest_point_on_segment/4, 560
neighbor_state/2, 1417
neighbor_state/3, 1418
neighbors/3, 605
nested_dictionary_protocol, 1152
new/1, 601, 808, 972, 1153, 1683
new/2, 601, 807, 972, 1745
new/3, 601, 775
new_line//0, 574
new_lines//0, 575
next/2, 1843
next/3, 1843
next/4, 452
next_impl/5, 470
next_occurrence/1, 1111
next_permutation/2, 1290
nextto/3, 1641, 1699
noc_metric, 174
node/6, 361
node/7, 411
node2_from_node4/2, 470
node_/6, 368
node_counter_/1, 343
node_id_/2, 344
node_path_/2, 368
nodebug/0, 314
nolog/3, 321
nologall/0, 321
non_blank//1, 576
non_blanks//1, 576
non_standard_predicate_call_/2, 1824
nonempty_subsequence/2, 1504
nonempty_subsequences/2, 1504
nor_metric, 175
normal/3, 1337

normal_element/2, 766
 normalize/2, 1750
 normalize_coordinate/2, 545
 normalize_date_time/2, 274
 normalize_polygon_orientation/3, 557
 normalize_range/2, 1670
 normalize_range/4, 1671
 normalize_scalar/2, 1672
 normalize_unit/2, 1671
 nosp/1, 317
 nosp/3, 318
 nosp/4, 319
 nospall/0, 319
 not64/2, 673
 not_excluded_file/3, 155
 not_excluded_file/4, 363
 note/1, 915
 note/2, 1277
 note/3, 1209
 notification/2, 843
 notification/3, 843
 notrace/0, 315
 now/3, 279
 nrmtree, 1157
 nth0/3, 1642, 1699
 nth0/4, 1642, 1699
 nth1/3, 1642, 1700
 nth1/4, 1643, 1700
 nth_combination/4, 190
 nth_permutation/3, 1291
 null/1, 813
 null_device_path/1, 1195
 number, 1658
 number//1, 584
 number_grammars(Format), 579
 number_of_edges/2, 607
 number_of_tests/1, 903
 number_of_vertices/2, 606
 number_string/2, 1491
 numberlist, 1662
 numberlistp, 1663
 numbervars/1, 1687
 numbervars/3, 1686

O

object_file_/2, 891
 object_predicate_called_/3, 1823
 object_wrapper_hook, 750
 object_wrapper_hook(Name,Relations), 753
 object_wrapper_hook(Protocol), 751
 observer, 325
 occurrences/2, 1644
 occurrences/3, 1644
 occurs/2, 1683

of/2, 1162
 of_expected/2, 500
 of_invalid/2, 1765
 of_invalids/2, 1766
 of_unexpected/2, 500
 of_valid/2, 1765
 offset/2, 1715
 offset/3, 1555, 1715
 offset/4, 1715
 omit_path_prefix/3, 367
 one_or_more//0, 588
 one_or_more//1, 587
 one_or_more//2, 586
 operating_system_machine/1, 1203
 operating_system_name/1, 1203
 operating_system_release/1, 1204
 operating_system_type/1, 1203
 optimize_rule_body/2, 255
 option/2, 1181
 option/3, 1181
 optional, 1161
 optional(Optional), 1166
 options, 1176
 options_protocol, 1177
 or/2, 514, 1170, 1778
 or64/3, 672
 or_else/2, 509, 1171, 1779
 or_else_call/2, 510, 1172, 1780
 or_else_fail/1, 511, 1173, 1780
 or_else_get/2, 510, 1172, 1779
 or_else_throw/1, 511, 1780
 or_else_throw/2, 515, 1173, 1781
 originator/1, 36
 orphaned/0, 1219
 orphaned/2, 1219
 os, 1183
 os_types, 1185
 osp, 1187
 out/1, 980
 out/2, 980
 out_degree/3, 593
 outdated/0, 1218
 outdated/1, 1218
 outdated/2, 1217
 outdated/4, 1216
 outdated/5, 1215
 output_edges/1, 362
 output externals/1, 359
 output_file/4, 360
 output_file_name/2, 409
 output_file_path/4, 364
 output_files/2, 360
 output_library/3, 359
 output_missing externals/1, 363

- output_node/6, 361
- output_rdirectory/3, 359
- output_rlibrary/3, 358
- output_schema/2, 1035
- output_sub_diagrams/1, 360
- overlapped_by/2, 778
- overlaps/2, 777

P

- pack_dependency/6, 1226
- pack_metadata/4, 1222
- pack_object/3, 1223
- pack_property/4, 1222
- pack_protocol, 1205
- package/1, 35
- packs, 1210
- packs_common, 1246
- packs_messages, 1258
- packs_specs_hook, 1259
- pad_md/4, 668
- pairing_heap(Order), 713
- pairing_heap_max, 714
- pairing_heap_min, 715
- pairs, 1674
- pairs_to_edges/2, 597
- params/2, 849
- parent/1, 740
- parent_stack_/1, 343
- parenthesis/2, 1615
- parents/1, 740
- parse/2, 59, 62, 64, 66, 68, 70, 76, 80, 824, 831, 838, 859, 1303, 1564, 1579, 1749, 1829, 1837
- parse/3, 59, 1303, 1829
- parse/4, 198
- parse/5, 198
- parse_all/2, 1838
- parse_domain/2, 1279
- parse_domain/3, 1279
- parse_error/1, 845
- parse_problem/2, 1280
- parse_problem/3, 1280
- partial_/1, 957, 959
- partial_map/4, 458
- partition/3, 497, 1761
- partition/4, 1054, 1398
- partition/5, 1645
- partition/6, 1054
- partition_body_literals/5, 256
- pass_info/1, 480
- pass_info/2, 479
- passed_/3, 945
- path/3, 607
- path_concat/3, 1191
- pdata_7bit//1, 1834
- pddl, 1278
- peek_back/2, 334
- peek_front/2, 334
- percentile/3, 1459
- permutation/2, 1285, 1330, 1645, 1701
- permutation/3, 1286
- permutation_index/3, 1291
- permutations, 1282
- permutations/2, 1285
- permutations/3, 1286
- permutations_protocol, 1284
- persist/3, 1377
- pid/1, 1189
- pin/0, 1249
- pin/1, 1249
- pinned/1, 1250
- plantuml_graph_language, 431
- plus/3, 1628
- point_in_polygon/2, 554
- point_to_polyline_distance/3, 560
- poisson/2, 1341
- polygon_area/2, 555
- polygon_bounding_box/2, 555
- polygon_centroid/2, 555
- polygon_orientation/2, 556
- polygon_perimeter/2, 563
- polygon_perimeter/3, 563
- polygons_intersect/2, 564
- polyline_length/2, 561
- polyline_length/3, 561
- polyline_resample/3, 563
- polyline_simplify/3, 562
- polyline_split_at_distance/4, 562
- pool_config/5, 28
- pop_back/3, 333
- pop_front/3, 333
- population, 1440
- port/5, 1296
- port_/5, 1298
- porter_stemmer(Representation), 1467
- portray_clause/1, 1005
- ports_profiler, 1293
- postorder/2, 443
- power/2, 1342
- power_sequence/4, 1629
- power_set/2, 1505
- powerset/2, 1405
- pp/1, 1830
- pp_string/1, 1832
- pprint/1, 1067
- pprint_clause/1, 1070
- pprint_clauses/1, 1070
- pq_insert/3, 636
- pred_info/3, 478

predicate/1, 340
 predicate/2, 293, 339, 1097
 predicate/3, 294, 1097
 predicate_breakpoint_/3, 307
 predicate_called_but_not_defined_/2, 1822
 predicate_coverage_/7, 896
 predicate_directive_suppression(Entity,Predicate,Direction,Integer,Source,PrintMutation), 1117
 predicate_entity_/4, 870
 predicate_info_pair_score_hook/4, 164
 predicate_info_score_hook/3, 164
 predicate_mode_score_hook/3, 163
 predicate_mode_score_hook/5, 163
 predicate_mutants/3, 1101
 predicate_mutants/4, 1101
 predicate_stratum/3, 263
 predicate_stratum_/3, 250
 predicates/2, 292
 predicates/3, 293
 predicates_in_stratum/2, 253
 predict/3, 128
 predict/4, 798, 861, 1148
 predict_probabilities/3, 3, 862, 1141, 1148, 1349
 predict_probabilities/4, 862, 1148
 preferred_mime_name/1, 92
 prefix/0, 1254
 prefix/1, 1254
 prefix/2, 1645, 1701
 prefix/3, 1646
 preorder/2, 443
 prepend/3, 1041
 previous/2, 1843
 previous/3, 1844
 previous/4, 451
 previous_impl/5, 470
 previous_permutation/2, 1290
 print_classifier/1, 130
 print_flags/0, 524, 529
 print_flags/1, 525
 print_goal_hook, 754
 print_message/3, 207
 print_message_token/4, 208
 print_message_tokens/3, 207
 print_mutation/3, 1107
 print_readme_file_path/1, 1255
 probe_mutation_happened_/0, 1103
 probing_/0, 1103
 process, 1299
 process_all/1, 146
 process_directory/2, 145
 process_entity/2, 144
 process_file/2, 145
 process_library/2, 146
 process_rdirectory/2, 145
 process_rlibrary/2, 146
 product/2, 1451, 1666
 product/3, 1405
 program_to_clauses/2, 1067
 progress/5, 1420
 prolog_module_hook(Module), 756
 prolog_module_hook(Integer,Source,PrintMutation), 1117
 prompts/1, 1022
 proper_prefix/2, 1646
 proper_prefix/3, 1647
 proper_subsequence/2, 1513
 proper_suffix/2, 1654
 proper_suffix/3, 1654
 property/1, 41
 proto_hierarchy, 737
 proto_hierarchyp, 738
 protobuf, 1302
 prove/2, 1805
 prove/3, 1805
 provides/2, 1271
 prune/3, 73
 prune/5, 72
 pseudo_random_protocol, 1319
 push_back/3, 333
 push_front/3, 333

Q

quartiles/4, 1460
 quasi_skipping_/0, 303
 query/1, 261
 query/2, 262
 question_hook/6, 210
 question_prompt_stream/4, 210
 queue, 1305
 queue_bind/4, 12
 queue_declare/3, 11
 queue_delete/3, 12
 queue_purge/2, 13
 queue_unbind/4, 13
 queuep, 1306
 quick_check/1, 907
 quick_check/2, 906
 quick_check/3, 906

R

random, 1321
 random(Algorithm), 1322
 random/1, 1327
 random/3, 1331
 random_combination/3, 191
 random_forest, 1348
 random_node/1, 1758
 random_permutation/2, 1292
 random_protocol, 1326

random_subsequence/2, 1518
random_tree/1, 1786
randomize/1, 1318, 1324
randseq/4, 1331
randset/4, 1332
range/2, 1453
rank_correlation/3, 1461
rbtree, 456
rd/1, 983
rd/2, 983
rd_list/2, 985
rd_list/3, 984
rd_noblock/1, 984
rd_noblock/2, 984
rdirectories/1, 879
rdirectories/2, 879
rdirectory/1, 139, 290, 353, 377, 880, 1817
rdirectory/2, 138, 290, 353, 377, 880, 1817
rdirectory/3, 353, 377
rdirectory_score/2, 143
reachable/3, 605
read_file, 1281
read_file/2, 235, 1282, 1592
read_file/3, 233, 1591
read_file_by_line/2, 238, 1595
read_file_by_line/3, 236, 1594
read_framed_message/2, 853
read_from_atom/2, 1526
read_from_chars/2, 1527
read_from_codes/2, 1528
read_message/2, 852
read_mime_types/2, 1079
read_only_device_path/1, 1196
read_stream/2, 235, 1593
read_stream/3, 234, 1591
read_stream_by_line/2, 238, 1596
read_stream_by_line/3, 237, 1595
read_term_from_atom/3, 1526
read_term_from_chars/3, 1526
read_term_from_chars/4, 1527
read_term_from_codes/3, 1528
read_term_from_codes/4, 1528
reader, 1350
readme/1, 1252
readme/2, 1252
readme_file_path/2, 1255
receive/3, 18, 1475
record_/3, 1366
recorda/2, 1363
recorda/3, 1362
recorded/2, 1364
recorded/3, 1364
recorded_database, 1360
recorded_database_core, 1361

recorded_linter_warning_/7, 944
recorded_warning_/4, 1002
recordz/2, 1364
recordz/3, 1363
redis, 1367
reference_/1, 1366
referenced_entity_/2, 397
referenced_logtalk_directory_/1, 387
referenced_logtalk_file_/1, 404
referenced_logtalk_library_/2, 422
referenced_module_/2, 397
referenced_predicate_/1, 439
referenced_prolog_directory_/1, 387
referenced_prolog_file_/1, 404
referenced_prolog_library_/2, 423
registries, 1261
registry_loader_hook, 1273
registry_protocol, 1274
relational_operator_replacement(Entity,Predicate,ClauseIndex,Occu
1118
relative_standard_deviation/2, 1457
relax_neighbors/7, 636
release/1, 26
release_date/1, 37
remember_included_directory/1, 386
remember_included_file/1, 403
remember_included_library/2, 421
remember_referenced_logtalk_directory/1, 386
remember_referenced_logtalk_file/1, 403
remember_referenced_logtalk_library/2, 421
remember_referenced_prolog_directory/1, 386
remember_referenced_prolog_file/1, 403
remember_referenced_prolog_library/2, 422
remove_directive_/2, 1826
remove_duplicates/2, 1647, 1701
remove_rule/1, 259
removeDependent/1, 329
rename/4, 1378
rename_file/2, 1200
replace/3, 1847
replace/5, 1040
replace_sub_atom/4, 1603
report_directory/3, 1099
report_entity/3, 1097
report_library/3, 1098
report_predicate/4, 1098
repository/1, 38
repository_branch/1, 38
repository_commit/1, 39
repository_commit_abbreviated/1, 39
repository_commit_author/1, 40
repository_commit_date/1, 39
repository_commit_message/1, 40
request/3, 842

request/4, 842
 rescale/3, 1672
 reset/0, 313, 361, 1000, 1080, 1115, 1248, 1296, 1793
 reset/1, 1296
 reset_counter/1, 970
 reset_counters/0, 970
 reset_flags/0, 523
 reset_flags/1, 523
 reset_genint/0, 535
 reset_genint/1, 535
 reset_gensym/0, 538
 reset_gensym/1, 538
 reset_monitor/0, 491
 reset_seed/0, 1318, 1324
 resize/2, 28
 resource_read/3, 1024
 resources/1, 1024
 response/3, 843
 restore/1, 1241
 restore/2, 1240
 restore_edb_facts/1, 251
 restore_idb_facts/1, 252
 restore_predicate_strata/1, 253
 restore_snapshot/6, 251
 restore_support_counts/1, 252
 restore_support_edges/1, 252
 result/2, 850
 retract_fact/1, 260
 reverse/2, 1647, 1702
 rewind/2, 1844
 rewind/3, 1844
 rfc_metric, 177
 rhumb_bearing/3, 547
 rhumb_destination_point/4, 548
 rhumb_distance/3, 547
 rhumb_midpoint/3, 548
 rlibraries/1, 876
 rlibraries/2, 876
 rlibrary/1, 140, 292, 351, 375, 877
 rlibrary/2, 140, 291, 350, 374, 876
 rlibrary_score/2, 142
 rol32/3, 666
 rol64/3, 672
 rollback/0, 260
 root_mean_squared_error/3, 1464
 ror32/3, 667
 route_distance/2, 564
 route_distance/3, 565
 route_distance/4, 565
 rpop/3, 1382
 rpush/4, 1381
 rule/2, 1795
 rule/3, 1795
 rule/4, 1795

rule_/3, 249
 rule_expansion(Mode), 1809
 rules/1, 262
 run/0, 901
 run/1, 902
 run/2, 902, 1414
 run/3, 1414
 run/4, 1415
 run_quick_check_tests/5, 913
 run_test_set/0, 913
 run_test_sets/1, 903
 run_tests/0, 913
 run_tests/1, 913
 running_test_sets_/0, 942
 runtime_type_/3, 1081

S

sadd/4, 1384
 same_instant/2, 276
 same_length/2, 1648, 1702
 same_length/3, 1648
 sample, 1441
 sampling_protocol, 1335
 sarif, 1388
 save/0, 1821
 save/1, 1239, 1711, 1820
 save/2, 1238, 1711
 save_dut1_entries/1, 1541, 1550
 save_edge/5, 362
 save_leap_seconds_entries/1, 1539, 1548
 sbom, 1391
 scalar_product/3, 1670
 scan_left/4, 1055
 scan_left_1/3, 1056
 scan_right/4, 1057
 scan_right_1/3, 1057
 scard/3, 1385
 scope_directive_replacement(Entity,Predicate,DirectiveIndex,Occurrence), 1119
 score/3, 798
 score_all/3, 799
 sdbm_32, 682
 sdbm_64, 683
 search/1, 1226
 secondary_header/2, 84
 secondary_header_flag/2, 82
 secondary_header_time/2, 84
 section_started_/1, 342
 seed_/2, 1318, 1325
 seen_/1, 1114
 select/3, 1328, 1405, 1648, 1702
 select/4, 471, 1329, 1649
 selectchk/3, 1406, 1649
 selectchk/4, 1649

selected_test_/1, 943
send/3, 1370
send/4, 1474
send_heartbeat/1, 20, 1477
sequence/2, 497, 1160, 1762
sequence/3, 1628
sequence/4, 1330, 1625, 1629
sequence/5, 1625
sequence_count/2, 83
sequence_flags/2, 82
sequence_grammars, 585
sequential_occurrences/2, 1643
sequential_occurrences/3, 1643
serve/3, 1311
server_accept/4, 1438
server_accept/5, 1437
server_close/1, 1438
server_name_/1, 1030
server_open/2, 1437
server_open/3, 1436
server_open/4, 1436
server_running_/1, 992
server_shutdown_/0, 992
server_socket_/1, 991
server_title_/1, 1030
server_version_/1, 1030
set, 1395
set(Type), 1396
set/1, 1792
set/3, 1039
set/4, 1331, 1371
set/5, 1039
set_binary_input/1, 922
set_binary_input/2, 922
set_binary_input/3, 921
set_binary_output/1, 932
set_binary_output/2, 931
set_binary_output/3, 931
set_element/2, 818
set_field/2, 807
set_flag_value/2, 522
set_flag_value/3, 523
set_monitor/4, 487
set_seed/1, 47, 1320
set_spy_point/4, 492
set_text_input/1, 918
set_text_input/2, 918
set_text_input/3, 917
set_text_output/1, 926
set_text_output/2, 925
set_text_output/3, 925
set_write_max_depth/1, 322
setp, 1399
setrange/5, 1372

setup/0, 914, 1248
sha1, 684
sha256, 686
sha256sum_command/1, 1256
sha3_224, 687
sha3_256, 689
sha3_384, 690
sha3_512, 691
shake128(OutputBytes), 692
shake256(OutputBytes), 694
shell, 1810
shell(Interpreters), 1812
shell/1, 1189
shell/2, 1189
shell_command/1, 476
shell_expansion(Mode), 1813
shl64/3, 673
short_flags/1, 194
shr64/3, 674
shrink/3, 46
shrink_sequence/3, 46
shriner/1, 45
shutdown_server/1, 979
sign//1, 584
simulated_annealing(Problem), 1410
simulated_annealing(Problem,RandomAlgorithm),
1411
simulated_annealing_protocol, 1416
single_byte_character_set(MaxCode), 108
singletons/2, 1686
siphash_2_4, 695
siphash_2_4(Key), 696
sismember/4, 1385
size/2, 456, 706, 1404
size_metric, 178
skewness/2, 1457
skipped_/1, 945
skipping_/0, 303
skipping_unleashed_/1, 303
sleep/1, 1205
sliding_window/3, 1517
smembers/3, 1385
snapshot_/6, 251
snowflakeid, 1421
snowflakeid(Representation,EpochMilliseconds,TimeUnitMilliseconds),
1422
snowflakeid_instagram, 1424
snowflakeid_instagram(Representation), 1425
snowflakeid_protocol, 1426
snowflakeid_sonyflake, 1428
snowflakeid_sonyflake(Representation), 1429
snowflakeid_twitter, 1430
snowflakeid_twitter(Representation), 1432
socket, 1433

softmax/2, 1673
 softmax/3, 1673
 software_heritage_identifier/1, 41
 sort/2, 1398, 1650
 sort/3, 1650
 sort/4, 1651
 sorted_median/3, 1446
 soundex/2, 1487
 soundex_match/2, 1487
 source_file_extension/1, 430
 space//0, 573
 spaces//0, 573
 spdx_license_schema_/1, 1395
 splaytree, 458
 split/3, 1603
 split/4, 1651
 split_string/4, 1494
 spy/1, 316
 spy/3, 317
 spy/4, 318
 spy_point/4, 492
 spy_point_/4, 489
 spying/1, 316
 spying/3, 318
 spying/4, 319
 squared_error_sum/6, 1448
 squares_and_cubes/6, 1445
 squares_and_hypers/6, 1445
 srem/4, 1384
 standard_cauchy/3, 1346
 standard_deviation/2, 1456
 standard_error/2, 1463
 standard_exponential/1, 1346
 standard_gamma/2, 1346
 standard_normal/1, 1347
 standard_t/2, 1345
 start/0, 1294, 1811
 start/2, 1027
 start/3, 1028
 start/4, 1028
 start/5, 1028
 start_redirect_to_file/2, 324
 started_by/2, 778
 starts/2, 778
 state_energy/2, 1418
 statistics, 1443
 statisticsp, 1449
 stats/1, 27
 stats/2, 1048
 stats/3, 1048
 status_file_/1, 1125
 stem/2, 1469
 stemmer_protocol, 1468
 stems/2, 1470
 stomp, 1471
 stop/0, 1294
 stop_condition/3, 1419
 stop_redirect_to_file/0, 324
 strata/1, 263
 strata_from_numbers/2, 253
 stream_position/1, 941
 stream_to_bytes/2, 1357
 stream_to_bytes/3, 1357
 stream_to_chars/2, 1356
 stream_to_chars/3, 1356
 stream_to_codes/2, 1355
 stream_to_codes/3, 1355
 stream_to_terms/2, 1356
 stream_to_terms/3, 1357
 streamvars, 971
 string(Representation), 1490
 string_chars/2, 1492
 string_codes/2, 1492
 string_concat/3, 1493
 string_distance(Representation), 1480
 string_length/2, 1493
 string_lower/2, 1494
 string_upper/2, 1494
 strlen/3, 1373
 strongly_connected_components/2, 595
 sub_diagram_/1, 401, 408, 419, 426
 sub_diagram_/2, 384, 391
 sub_directory/2, 147
 sub_library/2, 147
 sub_string/5, 1493
 subclass/1, 729
 subclasses/1, 730
 subject, 326
 sublist/2, 1651, 1703
 subprocess_coverage_hook, 1121
 subprocess_mutation_hook, 1123
 subscribe/4, 1474
 subsequence/2, 1502
 subsequence/3, 1503, 1652
 subsequence/4, 1652
 subsequence_at_indices/3, 1514
 subsequence_length/2, 1521
 subsequences, 1498
 subsequences/2, 1502
 subsequences/3, 1503
 subsequences_protocol, 1500
 subsequences_with_min_span/3, 1518
 subset/2, 1406
 subslices/2, 1517
 substitute/4, 1653
 subsumes/2, 1684
 subterm/2, 1684, 1830
 subtract/3, 1406, 1653, 1703

- subtract_duration/3, 270
- subunit_v1_output, 950
- subunit_v1_report, 951
- subunit_v2_output, 952
- subunit_v2_report, 954
- succ/2, 1628
- suffix/2, 1653, 1703
- suffix/3, 1654
- suffix_alias/2, 1080
- sum/2, 1451, 1666
- sum_of_squares/2, 1462
- sum_of_squares/4, 1447
- summary/1, 1000
- summary_/2, 889
- superclass/1, 730
- superclasses/1, 730
- supplier/1, 36
- support_count_/2, 250
- support_edge_/3, 250
- supported_archive/1, 1257
- supported_editor_url_scheme_prefix/1, 366
- supported_range/2, 1546
- supported_url_archive/1, 1257
- suppress_binary_output/0, 917
- suppress_goal_hook, 757
- suppress_text_output/0, 916
- suspend_monitor/0, 491
- swap/1, 513, 1778
- swap/2, 1329
- swap_consecutive/2, 1329
- syndiff/3, 1407
- symmetric_closure/2, 592

T

- tab//0, 574
- tabs//0, 574
- tai_minus_utc_for_tai_unix/2, 1544
- tail/2, 1507
- tail1/2, 1509
- tails/2, 1506
- tails1/2, 1508
- take/3, 1655
- take/4, 1655
- tap_output, 955
- tap_report, 957
- tar_command/1, 1257
- target_predicate/3, 1108
- target_predicate_clause_index/4, 1108
- target_predicate_directive/3, 1109
- target_predicate_directive_index/4, 1110
- target_scope_directive/3, 1108
- target_scope_directive_index/4, 1109
- target_uses_directive/3, 1110
- target_uses_directive_index/4, 1110

- tcb_minus_tdb_approx/3, 1544
- tcg_minus_tt_approx/3, 1543
- tdb_minus_tt_approx/3, 1543
- temporary_directory/1, 1195
- temporary_file_/1, 1524
- term, 1680
- term/2, 1390
- term/4, 1390
- term_expansion/2, 202
- term_io, 1522
- term_io_protocol, 1524
- termp, 1681
- terms_to_array/2, 815
- test/1, 903
- test/2, 942
- test/3, 942
- test_/2, 943
- test_count_/1, 957, 959
- text_file_assertion/3, 938
- text_file_assertion/4, 938
- text_input_assertion/2, 920
- text_input_assertion/3, 920
- text_output_assertion/2, 929
- text_output_assertion/3, 928
- text_output_assertion/4, 928
- text_output_contents/1, 930
- text_output_contents/2, 930
- text_output_contents/3, 929
- time, 277
- time_scales, 1535
- time_scales_data, 1536
- time_scales_protocol, 1545
- time_stamp/1, 1201
- time_string/3, 789
- time_type/2, 1714
- time_type/3, 1714
- time_type/4, 1713
- timeout, 1557
- timeout/1, 1069
- timep, 278
- timestamp/2, 1740
- timestamp/8, 1740
- timestamp_/6, 887, 891
- to_expected/1, 1782
- to_expected/2, 1175
- to_optional/1, 515, 1782
- today/3, 267
- tolerance_equal/4, 911, 1660
- toml, 1559
- toml(ObjectRepresentation,PairRepresentation,StringRepresentation), 1562
- toml(StringRepresentation), 1561
- toml_protocol, 1563
- tool/1, 720

- tool_call/3, 1034
 - tool_call/4, 1034
 - tool_diagnostics_common, 1566
 - tool_diagnostics_protocol, 1568
 - tools/0, 720
 - tools/1, 1033
 - toon, 1574
 - toon(ObjectRepresentation,PairRepresentation,StringRepresentation), 1576
 - toon(StringRepresentation), 1575
 - toon_protocol, 1578
 - top/3, 707
 - top_next/5, 708
 - topological_sort/2, 593
 - topological_sort/3, 621
 - touch/3, 1045
 - toychrdb, 1580
 - trace/0, 315
 - trace_event/2, 211
 - tracing_/0, 302
 - transitive_closure/2, 592
 - transitive_reduction/2, 622
 - transpose/2, 592, 1678
 - traverse/3, 497, 1160, 1763
 - treap_set, 1408
 - triangular/4, 1343
 - triggered_breakpoint_/4, 308
 - triggered_breakpoint_enabled_/2, 309
 - trim/2, 1496
 - trim/3, 1496
 - trim_left/2, 1496
 - trim_left/3, 1497
 - trim_right/2, 1497
 - trim_right/3, 1497
 - trimmed_mean/3, 1462
 - true/1, 812
 - truth_literal_flip(Entity,Predicate,ClauseIndex,Occurrence,PrintMutation), 1125
 - ts_findall_in_noblock/3, 996
 - ts_findall_rd_noblock/3, 996
 - ts_in/4, 994
 - ts_in_list/4, 995
 - ts_in_noblock/2, 994
 - ts_out/1, 994
 - ts_rd/4, 995
 - ts_rd_list/4, 996
 - ts_rd_noblock/2, 995
 - tsv, 1585
 - tsv(Header), 1587
 - tsv(Header,Comments), 1588
 - tsv_protocol, 1589
 - tt_minus_tai/2, 1542
 - ttl/3, 1376
 - tuple_/1, 993
 - tutor, 1599
 - tutor_explanations, 1600
 - two3tree, 460
 - tx_commit/1, 19
 - tx_rollback/1, 20
 - tx_select/1, 19
 - type, 1688
 - type/1, 81
 - type/2, 81
 - type/3, 1070, 1378
 - type_entity_/4, 870
 - tzdb_version/1, 1734
 - tzif, 1705
 - tzif_protocol, 1707
 - tzif_zone_ids, 1733
- ## U
- ulid, 1735
 - ulid(Representation), 1736
 - ulid_protocol, 1738
 - ulid_types, 1741
 - undefined/2, 107
 - undirected_graph_common, 613
 - unexpected/1, 507
 - unexpecteds/2, 496
 - uniform/1, 1343
 - uniform/3, 1342
 - uninstall/0, 1237
 - uninstall/1, 1236
 - uninstall/2, 1235
 - union/3, 462, 621, 1407
 - union/4, 1408, 1745
 - union_all/3, 1746
 - union_find, 1742
 - union_find_protocol, 1744
 - union_impl/3, 462
 - union_impl/4, 463
 - unix_to_date_time/2, 269
 - unknown_predicate_called_/2, 1823
 - unpin/0, 1250
 - unpin/1, 1250
 - unsafe_set_flag_value/2, 526
 - unsubscribe/3, 1475
 - unweighted_directed_graph, 618
 - unweighted_directed_graph(Dictionary), 619
 - unweighted_graph_common(Dictionary), 622
 - unweighted_graph_protocol, 624
 - unweighted_undirected_graph, 627
 - unweighted_undirected_graph(Dictionary), 628
 - unzip/2, 1842
 - update/0, 475, 1235, 1268
 - update/1, 326, 1234, 1267
 - update/2, 1233, 1266
 - update/3, 261, 449, 1232

update/4, 448
update/5, 448
update_impl/3, 471
update_in/4, 1155
update_in/5, 1155
update_target_predicate_clause_index_/2, 1111
update_target_predicate_directive_index_/2, 1113
update_target_scope_directive_index_/2, 1112
update_target_uses_directive_index_/2, 1113
upn_metric, 180
url(Representation), 1748
us_ascii, 109
user, 222
user_data/2, 83
uses_diagram, 432
uses_diagram(Format), 434
uses_directive_resource_deletion(Entity,Predicate,DirectiveIndex,Occurrence,PrintMutation), 1127
utc_date_time_to_instant/2, 1550
utc_to_local/3, 271
utc_to_local_tz/3, 283
utf_16_character_set(Endian), 110
utf_16be, 111
utf_16le, 113
utf_32_character_set(Endian), 114
utf_32be, 115
utf_32le, 116
utf_8, 117
utf_8_character_set, 119
uuid, 1752
uuid(Representation), 1753
uuid_max/1, 1758
uuid_nil/1, 1758
uuid_null/1, 1757
uuid_protocol, 1754
uuid_v1/2, 1756
uuid_v3/3, 1756
uuid_v4/1, 1756
uuid_v5/3, 1757
uuid_v7/1, 1757

V

valid/0, 193
valid/1, 776, 1465, 1684, 1704, 1749, 1773
valid/2, 1692
valid/3, 269, 280
valid_conversion/3, 1552
valid_coordinate/1, 545
valid_date/3, 792
valid_date_time/1, 275
valid_instant/1, 1551
valid_option/1, 1179
valid_options/1, 1180

valid_scale/1, 1546
valid_unicode_scalar/1, 87
valid_until_date/1, 37
validate/1, 529
validate/2, 859
validate/3, 528, 859
validate_type/1, 528
validated, 1759
validation, 1764
validation(Validation), 1770
valids/2, 1761
value/1, 1792
value/3, 1677
value_reference/2, 812
values/2, 454, 1677
variables/2, 1685
variance/2, 1458
variance_index/6, Occurrence, PrintMutation), 1127
variant/2, 909, 1685
varlist, 1693
varlistp, 1695
varnumbers/2, 1687
varnumbers/3, 1687
verify_commands_availability/0, 1248
version/1, 33
version/2, 81, 1047
version/6, 1208
versions/3, 1220
vertex_neighbors_to_edges/4, 597
vertices/2, 602
vincenty_distance/3, 547
void/1, 813
void_element/1, 765
von_mises/3, 1344

W

wait/2, 1300
waiting_/3, 993
wald/3, 1337
wall_time/1, 1202
warning/1, 1000
warning_sequence_/1, 1001
warnings/1, 1000
weakly_connected_components/2, 595
week_of_year_iso/2, 273
weekday/2, 274
weibull/3, 1342
weighted_directed_graph, 630
weighted_directed_graph(Dictionary), 632
weighted_graph_common(Dictionary), 633
weighted_graph_protocol, 637
weighted_mean/3, 1454
weighted_undirected_graph, 641
weighted_undirected_graph(Dictionary), 642

welcome/0, 1811
 wfind/3, 635
 when/2, 226
 whiledo/2, 1016
 white_space//0, 573
 white_spaces//0, 573
 windows_1250, 120
 windows_1251, 121
 windows_1252, 122
 windows_1253, 123
 windows_1254, 125
 windows_1257, 126
 wininsert_neighbor/4, 635
 with_connection/1, 27
 with_output_to/2, 1534
 within_distance/4, 552
 without//2, 588
 wmc_metric, 181
 wneighbors/3, 639
 word32_hex/2, 664
 word64_hex/2, 670
 word_bytes/4, 89
 working_directory/1, 1194
 wpairs_to_edges/2, 598
 wrapper, 1815
 wremove_neighbor/4, 635
 wremove_vertex_from_all/3, 636
 write_file/3, 239, 1597
 write_framed_message/2, 852
 write_max_depth/1, 321
 write_max_depth_/1, 305
 write_message/2, 851
 write_stream/3, 240, 1597
 write_term_to_atom/3, 1529
 write_term_to_chars/3, 1530
 write_term_to_chars/4, 1530
 write_term_to_codes/3, 1531
 write_term_to_codes/4, 1531
 write_to_atom/2, 1529
 write_to_chars/2, 1530
 write_to_codes/2, 1531
 write_to_file_hook(File), 758
 write_to_file_hook(File,Options), 760
 write_to_stream_hook(Stream), 761
 write_to_stream_hook(Stream,Options), 762
 wvertex_neighbors_to_edges/4, 598

X

xhtml11, 768
 xml, 1827
 xml_to_document/3, 1831
 xor64/3, 672
 xref_diagram, 435
 xref_diagram(Format), 436

xunit_net_v2_output, 960
 xunit_net_v2_report, 961
 xunit_output, 963
 xunit_report, 964

Y

yaml, 1835
 yaml_protocol, 1836

Z

z_normalization/2, 1458
 zadd/5, 1386
 zap_to_port_/1, 305
 zcard/3, 1387
 zero_or_more//0, 588
 zero_or_more//1, 587
 zero_or_more//2, 586
 zip/2, 1841
 zip/3, 515, 1174, 1781, 1842
 zip_at_index/4, 1851
 zipperp, 1840
 zlist, 1850
 zone/3, 1712
 zone_id_kind/2, 1734
 zones/1, 1713
 zones/2, 1713
 zrange/5, 1386
 zrank/4, 1387
 zrem/4, 1386
 zscore/4, 1388