
Logtalk APIs

Release v3.96.0

Paulo Moura

Nov 18, 2025

CONTENTS

1	Libraries	1
1.1	arbitrary	1
1.1.1	arbitrary	1
1.2	assertions	7
1.2.1	assertions	8
1.2.2	assertions(Mode)	9
1.2.3	assertions_messages	11
1.3	assignvars	12
1.3.1	assignvars	12
1.3.2	assignvarsp	13
1.4	base64	17
1.4.1	base64	17
1.4.2	base64url	19
1.5	cbor	21
1.5.1	cbor	21
1.5.2	cbor(StringRepresentation)	22
1.6	code_metrics	24
1.6.1	cc_metric	24
1.6.2	code_metric	26
1.6.3	code_metrics	39
1.6.4	code_metrics_messages	40
1.6.5	code_metrics_utilities	41
1.6.6	coupling_metric	46
1.6.7	dit_metric	48
1.6.8	doc_metric	49
1.6.9	halstead_metric	54
1.6.10	halstead_metric(Stroud)	55
1.6.11	noc_metric	57
1.6.12	nor_metric	58
1.6.13	size_metric	60
1.6.14	upn_metric	61
1.7	core	63
1.7.1	core_messages	63
1.7.2	expanding	64
1.7.3	forwarding	66
1.7.4	logtalk	67
1.7.5	monitoring	82
1.7.6	user	84
1.8	coroutining	86
1.8.1	coroutining	86

1.9	csv	89
1.9.1	csv	89
1.9.2	csv(Header,Separator,IgnoreQuotes)	90
1.9.3	csv_guess_questions	92
1.9.4	csv_protocol	93
1.10	dates	104
1.10.1	date	104
1.10.2	datep	105
1.10.3	time	109
1.10.4	timep	110
1.11	dead_code_scanner	112
1.11.1	dead_code_scanner	112
1.11.2	dead_code_scanner_messages	120
1.12	debug_messages	121
1.12.1	debug_messages	121
1.13	debugger	125
1.13.1	debugger	125
1.13.2	debugger_messages	135
1.13.3	debuggerp	136
1.13.4	dump_trace	148
1.14	dependents	150
1.14.1	observer	150
1.14.2	subject	151
1.15	diagrams	155
1.15.1	d2_graph_language	155
1.15.2	diagram(Format)	156
1.15.3	diagrams	179
1.15.4	diagrams(Format)	180
1.15.5	directory_dependency_diagram	191
1.15.6	directory_dependency_diagram(Format)	192
1.15.7	directory_diagram(Format)	194
1.15.8	directory_load_diagram	198
1.15.9	directory_load_diagram(Format)	199
1.15.10	dot_graph_language	201
1.15.11	entity_diagram	202
1.15.12	entity_diagram(Format)	204
1.15.13	file_dependency_diagram	207
1.15.14	file_dependency_diagram(Format)	209
1.15.15	file_diagram(Format)	211
1.15.16	file_load_diagram	214
1.15.17	file_load_diagram(Format)	216
1.15.18	graph_language_protocol	218
1.15.19	graph_language_registry	222
1.15.20	inheritance_diagram	223
1.15.21	inheritance_diagram(Format)	225
1.15.22	library_dependency_diagram	226
1.15.23	library_dependency_diagram(Format)	227
1.15.24	library_diagram(Format)	229
1.15.25	library_load_diagram	234
1.15.26	library_load_diagram(Format)	235
1.15.27	mermaid_graph_language	237
1.15.28	modules_diagram_support	238
1.15.29	uses_diagram	241
1.15.30	uses_diagram(Format)	242

1.15.31	xref_diagram	244
1.15.32	xref_diagram(Format)	245
1.16	dictionaries	248
1.16.1	avltree	248
1.16.2	bintree	250
1.16.3	dictionaryp	253
1.16.4	rbtree	265
1.17	dif	267
1.17.1	dif	267
1.18	doclet	269
1.18.1	doclet	269
1.19	edcg	271
1.19.1	edcg	272
1.20	events	275
1.20.1	after_event_registry	275
1.20.2	before_event_registry	276
1.20.3	event_registry	278
1.20.4	event_registryp	279
1.20.5	monitor	283
1.20.6	monitorp	284
1.21	expand_library_alias_paths	288
1.21.1	expand_library_alias_paths	288
1.22	expecteds	289
1.22.1	either	289
1.22.2	expected	292
1.22.3	expected(Expected)	297
1.23	fcube	305
1.23.1	fcube	305
1.24	flags	309
1.24.1	flags	309
1.24.2	flags_validator	316
1.25	format	318
1.25.1	format	318
1.26	genint	320
1.26.1	genint	320
1.26.2	genint_core	322
1.27	gensym	324
1.27.1	gensym	324
1.27.2	gensym_core	325
1.28	git	328
1.28.1	git	328
1.28.2	git_protocol	329
1.29	grammars	333
1.29.1	blank_grammars(Format)	333
1.29.2	ip_grammars(Format)	340
1.29.3	number_grammars(Format)	341
1.29.4	sequence_grammars	347
1.30	heaps	351
1.30.1	heap(Order)	351
1.30.2	heapp	352
1.30.3	maxheap	358
1.30.4	minheap	359
1.31	help	360
1.31.1	help	360

1.31.2	help_info_support	367
1.32	hierarchies	370
1.32.1	class_hierarchy	370
1.32.2	class_hierarchyp	371
1.32.3	hierarchyp	378
1.32.4	proto_hierarchy	381
1.32.5	proto_hierarchyp	383
1.33	hook_flows	385
1.33.1	hook_pipeline(Pipeline)	386
1.33.2	hook_set(Set)	387
1.34	hook_objects	388
1.34.1	backend_adapter_hook	389
1.34.2	default_workflow_hook	390
1.34.3	grammar_rules_hook	391
1.34.4	identity_hook	393
1.34.5	object_wrapper_hook	394
1.34.6	object_wrapper_hook(Protocol)	395
1.34.7	object_wrapper_hook(Name,Relations)	397
1.34.8	print_goal_hook	398
1.34.9	prolog_module_hook(Module)	400
1.34.10	suppress_goal_hook	401
1.34.11	write_to_file_hook(File)	402
1.34.12	write_to_file_hook(File,Options)	404
1.34.13	write_to_stream_hook(Stream)	405
1.34.14	write_to_stream_hook(Stream,Options)	407
1.35	html	408
1.35.1	html	408
1.35.2	html5	411
1.35.3	xhtml11	412
1.36	ids	413
1.36.1	ids	413
1.36.2	ids(Representation,Bytes)	415
1.37	intervals	416
1.37.1	interval	417
1.37.2	intervalp	418
1.38	iso8601	425
1.38.1	iso8601	425
1.39	issue_creator	436
1.39.1	issue_creator	436
1.40	java	438
1.40.1	java	438
1.40.2	java(Reference)	439
1.40.3	java(Reference,ReturnValue)	441
1.40.4	java_access_protocol	442
1.40.5	java_hook	445
1.40.6	java_utils_protocol	447
1.41	json	456
1.41.1	json	456
1.41.2	json(StringRepresentation)	457
1.41.3	json(ObjectRepresentation,PairRepresentation,StringRepresentation)	458
1.41.4	json_protocol	460
1.42	json_lines	462
1.42.1	json_lines	462
1.42.2	json_lines(StringRepresentation)	463

1.42.3	json_lines(ObjectRepresentation,PairRepresentation,StringRepresentation)	464
1.42.4	json_lines_protocol	466
1.43	lgtdoc	468
1.43.1	lgtdoc	468
1.43.2	lgtdoc_messages	471
1.43.3	lgtdocp	472
1.44	lgtunit	484
1.44.1	automation_report	484
1.44.2	coverage_report	485
1.44.3	lgtunit	487
1.44.4	lgtunit_messages	536
1.44.5	minimal_output	538
1.44.6	tap_output	539
1.44.7	tap_report	541
1.44.8	xunit_net_v2_output	543
1.44.9	xunit_net_v2_report	545
1.44.10	xunit_output	546
1.44.11	xunit_report	548
1.45	library	550
1.45.1	cloning	550
1.45.2	counters	551
1.45.3	streamvars	555
1.46	listing	558
1.46.1	listing	558
1.47	logging	561
1.47.1	logger	561
1.47.2	logging	563
1.47.3	loggingp	565
1.48	loops	569
1.48.1	loop	569
1.48.2	loopp	570
1.49	meta	576
1.49.1	meta	576
1.49.2	metap	577
1.50	meta_compiler	588
1.50.1	meta_compiler	588
1.51	metagol	590
1.51.1	metagol	590
1.51.2	metagol_example_protocol	597
1.52	mutations	599
1.52.1	default_atom_mutations	599
1.52.2	default_compound_mutations	601
1.52.3	default_float_mutations	602
1.52.4	default_integer_mutations	603
1.52.5	default_list_mutations	605
1.52.6	mutations	606
1.52.7	mutations_store	608
1.53	nested_dictionaries	611
1.53.1	navltree	611
1.53.2	nbintree	612
1.53.3	nested_dictionary_protocol	614
1.53.4	nrbtree	618
1.54	optionals	620
1.54.1	maybe	620

1.54.2	optional	622
1.54.3	optional(Optional)	626
1.55	options	634
1.55.1	options	634
1.55.2	options_protocol	635
1.56	os	641
1.56.1	os	642
1.56.2	os_types	643
1.56.3	osp	645
1.57	packs	664
1.57.1	pack_protocol	664
1.57.2	packs	668
1.57.3	packs_common	696
1.57.4	packs_messages	709
1.57.5	packs_specs_hook	710
1.57.6	registries	712
1.57.7	registry_loader_hook	724
1.57.8	registry_protocol	725
1.58	pddl_parser	729
1.58.1	pddl	729
1.58.2	read_file	731
1.59	ports_profiler	733
1.59.1	ports_profiler	733
1.60	queues	739
1.60.1	queue	739
1.60.2	queuep	741
1.61	random	747
1.61.1	backend_random	747
1.61.2	fast_random	749
1.61.3	pseudo_random_protocol	752
1.61.4	random	754
1.61.5	random_protocol	757
1.61.6	sampling_protocol	766
1.62	reader	779
1.62.1	reader	779
1.63	recorded_database	788
1.63.1	recorded_database	788
1.63.2	recorded_database_core	789
1.64	redis	795
1.64.1	redis	795
1.65	sets	798
1.65.1	set	798
1.65.2	set(Type)	800
1.65.3	setp	802
1.66	statistics	811
1.66.1	population	812
1.66.2	sample	813
1.66.3	statistics	814
1.66.4	statisticsp	817
1.67	term_io	828
1.67.1	term_io	828
1.67.2	term_io_protocol	830
1.68	timeout	840
1.68.1	timeout	840

1.69	toychr	842
1.69.1	toychrdb	843
1.70	tsv	848
1.70.1	tsv	848
1.70.2	tsv(Header)	849
1.70.3	tsv_protocol	851
1.71	tutor	860
1.71.1	tutor	860
1.72	types	862
1.72.1	atom	862
1.72.2	atomic	864
1.72.3	callable	865
1.72.4	character	866
1.72.5	characterp	868
1.72.6	comparingp	877
1.72.7	compound	880
1.72.8	difflist	881
1.72.9	float	884
1.72.10	integer	886
1.72.11	list	890
1.72.12	list(Type)	892
1.72.13	listp	893
1.72.14	natural	916
1.72.15	number	917
1.72.16	numberlist	921
1.72.17	numberlistp	923
1.72.18	pairs	933
1.72.19	term	938
1.72.20	termp	940
1.72.21	type	947
1.72.22	varlist	952
1.72.23	varlistp	954
1.73	ulid	964
1.73.1	ulid	964
1.73.2	ulid(Representation)	965
1.73.3	ulid_protocol	967
1.73.4	ulid_types	970
1.74	union_find	971
1.74.1	union_find	971
1.74.2	union_find_protocol	973
1.75	uuid	977
1.75.1	uuid	977
1.75.2	uuid(Representation)	978
1.75.3	uuid_protocol	980
1.76	verdi_neruda	982
1.76.1	a_star_interpreter(W)	983
1.76.2	benchmark_generators	984
1.76.3	best_first	985
1.76.4	bfs_interpreter	987
1.76.5	bup_interpreter	988
1.76.6	counter	989
1.76.7	databasep	993
1.76.8	debug_expansion(Mode)	995
1.76.9	demodb	997

1.76.10	dfs_interpreter	998
1.76.11	flatting	999
1.76.12	heuristic_expansion(Mode)	1001
1.76.13	iddfs_interpreter(Increment)	1002
1.76.14	interpreterp	1003
1.76.15	magic	1005
1.76.16	magic_expansion(Mode)	1007
1.76.17	rule_expansion(Mode)	1008
1.76.18	shell	1009
1.76.19	shell(Interpreters)	1011
1.76.20	shell_expansion(Mode)	1012
1.77	wrapper	1014
1.77.1	wrapper	1014
1.78	xml_parser	1026
1.78.1	xml	1026
1.79	zippers	1034
1.79.1	zipperp	1034
1.79.2	zlist	1044
2	Directories	1047
2.1	contributions/flags/	1049
2.2	contributions/iso8601/	1049
2.3	contributions/pddl_parser/	1049
2.4	contributions/verdi_neruda/	1049
2.5	contributions/xml_parser/	1049
2.6	core/	1049
2.7	library/	1049
2.8	library/arbitrary/	1049
2.9	library/assignvars/	1049
2.10	library/base64/	1049
2.11	library/cbor/	1049
2.12	library/coroutining/	1049
2.13	library/csv/	1049
2.14	library/dates/	1049
2.15	library/dependents/	1049
2.16	library/dictionaries/	1049
2.17	library/dif/	1049
2.18	library/edcg/	1049
2.19	library/events/	1049
2.20	library/expand_library_alias_paths/	1049
2.21	library/expecteds/	1049
2.22	library/format/	1049
2.23	library/genint/	1049
2.24	library/gensym/	1049
2.25	library/git/	1049
2.26	library/grammars/	1049
2.27	library/heaps/	1049
2.28	library/hierarchies/	1049
2.29	library/hook_flows/	1049
2.30	library/hook_objects/	1049
2.31	library/html/	1049
2.32	library/ids/	1049
2.33	library/intervals/	1049
2.34	library/java/	1049

2.35	library/json/	1049
2.36	library/json_lines/	1049
2.37	library/listing/	1049
2.38	library/logging/	1049
2.39	library/loops/	1049
2.40	library/meta/	1049
2.41	library/meta_compiler/	1049
2.42	library/mutations/	1049
2.43	library/nested_dictionaries/	1049
2.44	library/optionals/	1049
2.45	library/options/	1049
2.46	library/os/	1049
2.47	library/queues/	1049
2.48	library/random/	1049
2.49	library/reader/	1049
2.50	library/recorded_database/	1049
2.51	library/redis/	1049
2.52	library/sets/	1049
2.53	library/statistics/	1049
2.54	library/term_io/	1049
2.55	library/timeout/	1049
2.56	library/tsv/	1049
2.57	library/types/	1049
2.58	library/ulid/	1049
2.59	library/union_find/	1049
2.60	library/uuid/	1049
2.61	library/zippers/	1049
2.62	ports/fcube/	1049
2.63	ports/metagol/	1049
2.64	ports/toychr/	1049
2.65	tools/assertions/	1049
2.66	tools/code_metrics/	1049
2.67	tools/dead_code_scanner/	1049
2.68	tools/debug_messages/	1049
2.69	tools/debugger/	1049
2.70	tools/diagrams/	1049
2.71	tools/doclet/	1049
2.72	tools/help/	1049
2.73	tools/issue_creator/	1049
2.74	tools/lgtdoc/	1049
2.75	tools/lgtunit/	1049
2.76	tools/packs/	1049
2.77	tools/ports_profiler/	1049
2.78	tools/tutor/	1049
2.79	tools/wrapper/	1049
3	Entities	1051
3.1	Categories	1051
3.2	Objects	1051
3.3	Protocols	1051
4	Predicates	1053
4.1	(/)/2	1053
4.2	(//)/2	1053

4.3	(<)/2	1053
4.4	(<=)/2	1053
4.5	(==)/2	1053
4.6	(=<)/2	1054
4.7	(=>)/2	1054
4.8	(=\=)/2	1054
4.9	=~/2	1054
4.10	(>)/2	1054
4.11	(>=)/2	1054
4.12	absolute_file_name/2	1054
4.13	activate_debug_handler/1	1054
4.14	activate_monitor/0	1055
4.15	active_debug_handler/1	1055
4.16	add/1	1055
4.17	add/2	1055
4.18	add/3	1055
4.19	addDependent/1	1055
4.20	after/2	1055
4.21	after/3	1055
4.22	all/0	1056
4.23	all/1	1056
4.24	all_files/0	1056
4.25	all_files/1	1056
4.26	all_libraries/0	1056
4.27	all_libraries/1	1056
4.28	all_score/1	1057
4.29	ancestor/1	1057
4.30	ancestors/1	1057
4.31	apis/0	1057
4.32	apis/1	1057
4.33	append/2	1057
4.34	append/3	1057
4.35	apply/2	1057
4.36	apply/4	1058
4.37	approximately_equal/2	1058
4.38	approximately_equal/3	1058
4.39	arbitrary/1	1058
4.40	arbitrary/2	1058
4.41	archive/1	1058
4.42	arithmetic_mean/2	1058
4.43	array_list/2	1058
4.44	array_to_list/2	1059
4.45	array_to_terms/2	1059
4.46	array_to_terms/3	1059
4.47	as_curly_bracketed/2	1059
4.48	as_dictionary/2	1059
4.49	as_difflist/2	1059
4.50	as_heap/2	1059
4.51	as_list/2	1059
4.52	as_nested_dictionary/2	1060
4.53	as_set/2	1060
4.54	ask_question/5	1060
4.55	assertion/1	1060
4.56	assertion/2	1060

4.57	assignable/1	1060
4.58	assignable/2	1060
4.59	available/0	1060
4.60	available/1	1061
4.61	available/2	1061
4.62	average/2	1061
4.63	average_deviation/3	1061
4.64	before/2	1061
4.65	before/3	1061
4.66	bench_goal/1	1061
4.67	benchmark/2	1061
4.68	benchmark/3	1062
4.69	benchmark/4	1062
4.70	benchmark_reified/3	1062
4.71	bernoulli/2	1062
4.72	beta/3	1062
4.73	between/3	1062
4.74	between/4	1062
4.75	binomial/3	1062
4.76	bit//1	1063
4.77	bits//1	1063
4.78	blank//0	1063
4.79	blanks//0	1063
4.80	body_pred/1	1063
4.81	branch/2	1063
4.82	built_in_directive/4	1063
4.83	built_in_flag/2	1063
4.84	built_in_method/4	1064
4.85	built_in_non_terminal/4	1064
4.86	built_in_predicate/4	1064
4.87	calendar_month/3	1064
4.88	call_with_timeout/2	1064
4.89	call_with_timeout/3	1064
4.90	cat/2	1064
4.91	change_directory/1	1064
4.92	changed/0	1065
4.93	changed/1	1065
4.94	chebyshev_distance/3	1065
4.95	chebyshev_norm/2	1065
4.96	check/1	1065
4.97	check/2	1065
4.98	check/3	1065
4.99	check_option/1	1065
4.100	check_options/1	1066
4.101	chi_squared/2	1066
4.102	chr_is/2	1066
4.103	chr_no_spy/1	1066
4.104	chr_nospy/0	1066
4.105	chr_notrace/0	1066
4.106	chr_option/2	1066
4.107	chr_spy/1	1066
4.108	chr_trace/0	1067
4.109	circular_uniform_cartesian/3	1067
4.110	circular_uniform_polar/3	1067

4.111 class/1	1067
4.112 classes/1	1067
4.113 clause/5	1067
4.114 clause_location/6	1067
4.115 clean/0	1067
4.116 clean/1	1068
4.117 clean/2	1068
4.118 clone/1	1068
4.119 clone/3	1068
4.120 clone/4	1068
4.121 coefficient_of_variation/2	1068
4.122 command_line_arguments/1	1068
4.123 commit_author/2	1068
4.124 commit_date/2	1069
4.125 commit_hash/2	1069
4.126 commit_hash_abbreviated/2	1069
4.127 commit_log/3	1069
4.128 commit_message/2	1069
4.129 compile_aux_clauses/1	1069
4.130 compile_predicate_heads/4	1069
4.131 compile_predicate_indicators/3	1069
4.132 completion/2	1070
4.133 completions/2	1070
4.134 connect/1	1070
4.135 connect/3	1070
4.136 console/1	1070
4.137 contains/2	1070
4.138 control//0	1070
4.139 control_construct/4	1070
4.140 controls//0	1071
4.141 copy_file/2	1071
4.142 counter/2	1071
4.143 cover/1	1071
4.144 cpu_time/1	1071
4.145 current/2	1071
4.146 data/0	1071
4.147 data/1	1071
4.148 data/2	1072
4.149 date/4	1072
4.150 date/5	1072
4.151 date/6	1072
4.152 date/7	1072
4.153 date_string/3	1072
4.154 date_time/7	1072
4.155 days_in_month/3	1072
4.156 deactivate_debug_handler/0	1073
4.157 debug/0	1073
4.158 debug_handler/1	1073
4.159 debug_handler/3	1073
4.160 debugging/0	1073
4.161 debugging/1	1073
4.162 decide/1	1073
4.163 decide/2	1073
4.164 decode_exception/2	1074

4.165	decode_exception/3	1074
4.166	decompile_predicate_heads/4	1074
4.167	decompile_predicate_indicators/4	1074
4.168	decompose_file_name/3	1074
4.169	decompose_file_name/4	1074
4.170	decrement_counter/1	1074
4.171	default_option/1	1074
4.172	default_options/1	1075
4.173	define_log_file/2	1075
4.174	defined/4	1075
4.175	defined_flag/6	1075
4.176	del_monitors/0	1075
4.177	del_monitors/4	1075
4.178	del_spy_points/4	1075
4.179	delete/0	1075
4.180	delete/1	1076
4.181	delete/2	1076
4.182	delete/3	1076
4.183	delete/4	1076
4.184	delete_all_after/2	1076
4.185	delete_all_after_and_unzip/2	1076
4.186	delete_all_before/2	1076
4.187	delete_all_before_and_unzip/2	1076
4.188	delete_and_next/2	1077
4.189	delete_and_previous/2	1077
4.190	delete_and_unzip/2	1077
4.191	delete_directory/1	1077
4.192	delete_directory_and_contents/1	1077
4.193	delete_directory_contents/1	1077
4.194	delete_file/1	1077
4.195	delete_in/4	1077
4.196	delete_matches/3	1078
4.197	delete_max/4	1078
4.198	delete_min/4	1078
4.199	dependents/1	1078
4.200	dependents/2	1078
4.201	dependents/3	1078
4.202	depth/2	1078
4.203	descendant/1	1078
4.204	descendant_class/1	1079
4.205	descendant_classes/1	1079
4.206	descendant_instance/1	1079
4.207	descendant_instances/1	1079
4.208	descendants/1	1079
4.209	describe/1	1079
4.210	describe/2	1079
4.211	description/1	1079
4.212	deterministic/1	1080
4.213	deterministic/2	1080
4.214	diagram_description/1	1080
4.215	diagram_name_suffix/1	1080
4.216	dif/1	1080
4.217	dif/2	1080
4.218	digit/1	1080

4.219	digits//1	1080
4.220	directories/1	1081
4.221	directories/2	1081
4.222	directories/3	1081
4.223	directory/1	1081
4.224	directory/2	1081
4.225	directory/3	1082
4.226	directory_exists/1	1082
4.227	directory_files/2	1082
4.228	directory_files/3	1082
4.229	directory_score/2	1082
4.230	dirichlet/2	1082
4.231	disable/1	1082
4.232	disable/2	1082
4.233	disable_logging/1	1083
4.234	disconnect/1	1083
4.235	disjoint/2	1083
4.236	disjoint_sets/2	1083
4.237	doc_goal/1	1083
4.238	dot//1	1083
4.239	dowhile/2	1083
4.240	drop/3	1083
4.241	during/2	1084
4.242	easter_day/3	1084
4.243	edge/6	1084
4.244	edge_case/2	1084
4.245	either/3	1084
4.246	empty/1	1084
4.247	enable/1	1085
4.248	enable/2	1085
4.249	enable_logging/1	1085
4.250	enabled/1	1085
4.251	enabled/2	1085
4.252	ensure_directory/1	1085
4.253	ensure_file/1	1085
4.254	entity/1	1085
4.255	entity/2	1086
4.256	entity_info_pair_score_hook/3	1086
4.257	entity_info_score_hook/2	1086
4.258	entity_predicates_weights_hook/2	1086
4.259	entity_prefix/2	1086
4.260	entity_score/2	1086
4.261	enumerate/2	1086
4.262	environment_variable/2	1086
4.263	epsilon/1	1087
4.264	equal/2	1087
4.265	erase/1	1087
4.266	essentially_equal/3	1087
4.267	euclidean_distance/3	1087
4.268	euclidean_norm/2	1087
4.269	exclude/3	1087
4.270	execution_context/7	1087
4.271	expand_library_path/2	1088
4.272	expected/1	1088

4.273	expecteds/2	1088
4.274	explain//1	1088
4.275	exponential/2	1088
4.276	extension/1	1088
4.277	extensions/1	1088
4.278	failed_test_reason//1	1088
4.279	false/1	1089
4.280	fcube/0	1089
4.281	file/1	1089
4.282	file/2	1089
4.283	file_exists/1	1089
4.284	file_footer/3	1089
4.285	file_header/3	1090
4.286	file_modification_time/2	1090
4.287	file_permission/2	1090
4.288	file_score/2	1090
4.289	file_size/2	1090
4.290	file_to_bytes/2	1090
4.291	file_to_bytes/3	1090
4.292	file_to_chars/2	1090
4.293	file_to_chars/3	1091
4.294	file_to_codes/2	1091
4.295	file_to_codes/3	1091
4.296	file_to_terms/2	1091
4.297	file_to_terms/3	1091
4.298	file_type_extension/2	1091
4.299	files/1	1091
4.300	files/2	1092
4.301	files/3	1092
4.302	filter/2	1092
4.303	find/4	1092
4.304	find/5	1092
4.305	findall_member/4	1092
4.306	findall_member/5	1092
4.307	finished_by/2	1093
4.308	finishes/2	1093
4.309	fisher/3	1093
4.310	flag_group_chk/1	1093
4.311	flag_groups/1	1093
4.312	flat_map/2	1093
4.313	flatten/2	1093
4.314	float//1	1093
4.315	fold_left/4	1094
4.316	fold_left_1/3	1094
4.317	fold_right/4	1094
4.318	fold_right_1/3	1094
4.319	fordownto/3	1094
4.320	fordownto/4	1094
4.321	fordownto/5	1094
4.322	foreach/3	1094
4.323	foreach/4	1095
4.324	format/2	1095
4.325	format/3	1095
4.326	format_entity_score//2	1095

4.327	format_object/1	1095
4.328	format_to_atom/3	1095
4.329	format_to_chars/3	1095
4.330	format_to_chars/4	1095
4.331	format_to_codes/3	1096
4.332	format_to_codes/4	1096
4.333	forto/3	1096
4.334	forto/4	1096
4.335	forto/5	1096
4.336	forward/1	1096
4.337	forward/2	1096
4.338	forward/3	1096
4.339	fractile/3	1097
4.340	freeze/2	1097
4.341	from_generator/2	1097
4.342	from_generator/3	1097
4.343	from_generator/4	1097
4.344	from_goal/2	1097
4.345	from_goal/3	1097
4.346	from_goal/4	1098
4.347	frozen/2	1098
4.348	full_device_path/1	1098
4.349	func_test/3	1098
4.350	functional/0	1098
4.351	gamma/3	1098
4.352	generate/1	1098
4.353	generate/2	1098
4.354	generate/8	1099
4.355	genint/2	1099
4.356	gensym/2	1099
4.357	geometric/2	1099
4.358	geometric_mean/2	1099
4.359	get/1	1099
4.360	get_field/2	1099
4.361	get_flag_value/2	1099
4.362	get_seed/1	1100
4.363	gnu/0	1100
4.364	goal_expansion/2	1100
4.365	graph_footer/5	1100
4.366	graph_header/5	1100
4.367	ground/1	1100
4.368	group_by_key/2	1100
4.369	group_consecutive_by_key/2	1100
4.370	group_sorted_by_key/2	1101
4.371	guess_arity/2	1101
4.372	guess_separator/2	1101
4.373	gumbel/3	1101
4.374	hamming_distance/3	1101
4.375	handbook/0	1101
4.376	handbook/1	1101
4.377	harmonic_mean/2	1101
4.378	head/2	1102
4.379	head_pred/1	1102
4.380	help/0	1102

4.381 hex_digit//1	1102
4.382 hex_digits//1	1102
4.383 home/1	1102
4.384 hypergeometric/4	1102
4.385 ibk/3	1102
4.386 if_empty/1	1103
4.387 if_expected/1	1103
4.388 if_expected_or_else/2	1103
4.389 if_present/1	1103
4.390 if_present_or_else/2	1103
4.391 if_unexpected/1	1103
4.392 include/3	1103
4.393 increase/1	1103
4.394 increment/0	1104
4.395 increment_counter/1	1104
4.396 init/0	1104
4.397 init_log_file/2	1104
4.398 inorder/2	1104
4.399 insert/3	1104
4.400 insert/4	1104
4.401 insert_after/3	1104
4.402 insert_all/3	1105
4.403 insert_before/3	1105
4.404 insert_in/4	1105
4.405 install/1	1105
4.406 install/2	1105
4.407 install/3	1105
4.408 install/4	1105
4.409 installed/0	1105
4.410 installed/1	1106
4.411 installed/3	1106
4.412 installed/4	1106
4.413 instance/1	1106
4.414 instance/2	1106
4.415 instances/1	1106
4.416 integer//1	1106
4.417 internal_os_path/2	1106
4.418 intersect/2	1107
4.419 intersection/2	1107
4.420 intersection/3	1107
4.421 intersection/4	1107
4.422 invoke/1	1107
4.423 invoke/2	1107
4.424 ipv4//1	1107
4.425 ipv6//1	1107
4.426 is_absolute_file_name/1	1108
4.427 is_alpha/1	1108
4.428 is_alphanumeric/1	1108
4.429 is_ascii/1	1108
4.430 is_bin_digit/1	1108
4.431 is_control/1	1108
4.432 is_dec_digit/1	1108
4.433 is_empty/0	1108
4.434 is_end_of_line/1	1109

4.435	is_expected/0	1109
4.436	is_false/1	1109
4.437	is_hex_digit/1	1109
4.438	is_layout/1	1109
4.439	is_letter/1	1109
4.440	is_lower_case/1	1109
4.441	is_newline/1	1109
4.442	is_null/1	1110
4.443	is_object/1	1110
4.444	is_octal_digit/1	1110
4.445	is_period/1	1110
4.446	is_present/0	1110
4.447	is_punctuation/1	1110
4.448	is_quote/1	1110
4.449	is_true/1	1110
4.450	is_unexpected/0	1111
4.451	is_upper_case/1	1111
4.452	is_void/1	1111
4.453	is_vowel/1	1111
4.454	is_white_space/1	1111
4.455	iterator_element/2	1111
4.456	join/3	1111
4.457	join_all/3	1111
4.458	jump/3	1112
4.459	jump_all/3	1112
4.460	jump_all_block/3	1112
4.461	key/2	1112
4.462	keys/2	1112
4.463	keys_values/3	1112
4.464	keysort/2	1112
4.465	kurtosis/2	1112
4.466	language_object/2	1113
4.467	last/2	1113
4.468	leaf/1	1113
4.469	leaf_class/1	1113
4.470	leaf_classes/1	1113
4.471	leaf_instance/1	1113
4.472	leaf_instances/1	1113
4.473	leap_year/1	1113
4.474	learn/0	1114
4.475	learn/1	1114
4.476	learn/2	1114
4.477	learn/3	1114
4.478	learn_seq/2	1114
4.479	learn_with_timeout/4	1114
4.480	leash/1	1114
4.481	leashing/1	1114
4.482	least_common_multiple/2	1115
4.483	leaves/1	1115
4.484	length/2	1115
4.485	libraries/1	1115
4.486	libraries/2	1115
4.487	libraries/3	1115
4.488	library/0	1116

4.489	library/1	1116
4.490	library/2	1116
4.491	library_score/2	1116
4.492	license/1	1116
4.493	line_to_chars/2	1116
4.494	line_to_chars/3	1117
4.495	line_to_codes/2	1117
4.496	line_to_codes/3	1117
4.497	lint/0	1117
4.498	lint/1	1117
4.499	lint/2	1117
4.500	list/0	1117
4.501	list_to_array/2	1117
4.502	listing/0	1118
4.503	listing/1	1118
4.504	loaded_file/1	1118
4.505	loaded_file_property/2	1118
4.506	loaded_files_topological_sort/1	1118
4.507	loaded_files_topological_sort/2	1118
4.508	log/3	1118
4.509	log_event/2	1118
4.510	log_file/2	1119
4.511	logging/1	1119
4.512	logging/3	1119
4.513	logistic/3	1119
4.514	lognormal/3	1119
4.515	logseries/2	1119
4.516	logtalk_packs/0	1119
4.517	logtalk_packs/1	1119
4.518	lookup/2	1120
4.519	lookup/3	1120
4.520	lookup_in/3	1120
4.521	lower_upper/2	1120
4.522	magic/2	1120
4.523	magicise/4	1120
4.524	make_directory/1	1120
4.525	make_directory_path/1	1120
4.526	make_set/3	1121
4.527	man/1	1121
4.528	manhattan_distance/3	1121
4.529	manhattan_norm/2	1121
4.530	manuals/0	1121
4.531	map/2	1121
4.532	map/3	1121
4.533	map/4	1122
4.534	map/5	1122
4.535	map/6	1122
4.536	map/7	1122
4.537	map/8	1122
4.538	map_element/2	1122
4.539	map_reduce/5	1122
4.540	max/2	1122
4.541	max/3	1123
4.542	max_clauses/1	1123

4.543	max_inv_preds/1	1123
4.544	max_size/1	1123
4.545	maybe/0	1123
4.546	maybe/1	1123
4.547	maybe/2	1123
4.548	maybe_call/1	1123
4.549	maybe_call/2	1124
4.550	mean_deviation/2	1124
4.551	median/2	1124
4.552	median_deviation/2	1124
4.553	meets/2	1124
4.554	member/2	1124
4.555	memberchk/2	1124
4.556	merge/3	1125
4.557	message_hook/4	1125
4.558	message_prefix_file/6	1125
4.559	message_prefix_stream/4	1125
4.560	message_tokens//2	1125
4.561	met_by/2	1125
4.562	meta_type/3	1125
4.563	metarule/6	1125
4.564	metarule_next_id/1	1126
4.565	min/2	1126
4.566	min/3	1126
4.567	min_clauses/1	1126
4.568	min_max/3	1126
4.569	modes/2	1126
4.570	module_property/2	1126
4.571	monitor/1	1127
4.572	monitor/4	1127
4.573	monitor_activated/0	1127
4.574	monitored/1	1127
4.575	monitors/1	1127
4.576	msort/2	1127
4.577	msort/3	1127
4.578	mutation/3	1127
4.579	name/1	1128
4.580	name_of_day/3	1128
4.581	name_of_month/3	1128
4.582	natural//1	1128
4.583	new/1	1128
4.584	new/2	1128
4.585	new/3	1128
4.586	new_line//0	1129
4.587	new_lines//0	1129
4.588	next/2	1129
4.589	next/3	1129
4.590	next/4	1129
4.591	nextto/3	1129
4.592	node/7	1129
4.593	nodebug/0	1129
4.594	nolog/3	1130
4.595	nologall/0	1130
4.596	non_blank//1	1130

4.597 non_blanks//1	1130
4.598 normal/3	1130
4.599 normal_element/2	1130
4.600 normalize_range/2	1130
4.601 normalize_range/4	1130
4.602 normalize_scalar/2	1131
4.603 normalize_unit/2	1131
4.604 nospy/1	1131
4.605 nospy/3	1131
4.606 nospy/4	1131
4.607 nospyall/0	1131
4.608 note/2	1131
4.609 note/3	1131
4.610 notrace/0	1132
4.611 now/3	1132
4.612 nth0/3	1132
4.613 nth0/4	1132
4.614 nth1/3	1132
4.615 nth1/4	1132
4.616 null/1	1132
4.617 null_device_path/1	1133
4.618 number//1	1133
4.619 number_of_tests/1	1133
4.620 numbervars/1	1133
4.621 numbervars/3	1133
4.622 occurrences/2	1133
4.623 occurrences/3	1133
4.624 occurs/2	1133
4.625 of/2	1134
4.626 of_expected/2	1134
4.627 of_unexpected/2	1134
4.628 one_or_more//0	1134
4.629 one_or_more//1	1134
4.630 one_or_more//2	1134
4.631 operating_system_machine/1	1134
4.632 operating_system_name/1	1134
4.633 operating_system_release/1	1135
4.634 operating_system_type/1	1135
4.635 option/2	1135
4.636 option/3	1135
4.637 or/2	1135
4.638 or_else/2	1135
4.639 or_else_call/2	1135
4.640 or_else_fail/1	1135
4.641 or_else_get/2	1136
4.642 or_else_throw/1	1136
4.643 or_else_throw/2	1136
4.644 orphaned/0	1136
4.645 orphaned/2	1136
4.646 outdated/0	1136
4.647 outdated/1	1136
4.648 outdated/2	1136
4.649 outdated/4	1137
4.650 outdated/5	1137

4.651	output_file_name/2	1137
4.652	overlapped_by/2	1137
4.653	overlaps/2	1137
4.654	parent/1	1137
4.655	parenthesis/2	1137
4.656	parents/1	1137
4.657	parse/2	1138
4.658	parse/3	1138
4.659	parse_domain/2	1138
4.660	parse_domain/3	1138
4.661	parse_problem/2	1138
4.662	parse_problem/3	1138
4.663	partial_map/4	1138
4.664	partition/3	1139
4.665	partition/4	1139
4.666	partition/5	1139
4.667	partition/6	1139
4.668	path_concat/3	1139
4.669	permutation/2	1139
4.670	pid/1	1139
4.671	pin/0	1139
4.672	pin/1	1140
4.673	pinned/1	1140
4.674	plus/3	1140
4.675	poisson/2	1140
4.676	port/5	1140
4.677	portray_clause/1	1140
4.678	postorder/2	1140
4.679	power/2	1140
4.680	power_sequence/4	1141
4.681	powerset/2	1141
4.682	pp/1	1141
4.683	pprint/1	1141
4.684	predicate/2	1141
4.685	predicate_info_pair_score_hook/4	1141
4.686	predicate_info_score_hook/3	1141
4.687	predicate_mode_score_hook/3	1141
4.688	predicate_mode_score_hook/5	1142
4.689	predicates/2	1142
4.690	prefix/0	1142
4.691	prefix/1	1142
4.692	prefix/2	1142
4.693	prefix/3	1142
4.694	preorder/2	1142
4.695	previous/2	1142
4.696	previous/3	1143
4.697	previous/4	1143
4.698	print_flags/0	1143
4.699	print_flags/1	1143
4.700	print_message/3	1143
4.701	print_message_token/4	1143
4.702	print_message_tokens/3	1143
4.703	product/2	1143
4.704	product/3	1144

4.705	program_to_clauses/2	1144
4.706	proper_prefix/2	1144
4.707	proper_prefix/3	1144
4.708	proper_suffix/2	1144
4.709	proper_suffix/3	1144
4.710	prove/2	1144
4.711	prove/3	1144
4.712	provides/2	1145
4.713	question_hook/6	1145
4.714	question_prompt_stream/4	1145
4.715	quick_check/1	1145
4.716	quick_check/2	1145
4.717	quick_check/3	1145
4.718	random/1	1145
4.719	random/3	1145
4.720	random_node/1	1146
4.721	random_tree/1	1146
4.722	randomize/1	1146
4.723	randseq/4	1146
4.724	randset/4	1146
4.725	range/2	1146
4.726	rdirectories/1	1146
4.727	rdirectories/2	1146
4.728	rdirectory/1	1147
4.729	rdirectory/2	1147
4.730	rdirectory/3	1147
4.731	rdirectory_score/2	1147
4.732	read_file/2	1147
4.733	read_file/3	1148
4.734	read_file_by_line/2	1148
4.735	read_file_by_line/3	1148
4.736	read_from_atom/2	1148
4.737	read_from_chars/2	1148
4.738	read_from_codes/2	1148
4.739	read_only_device_path/1	1148
4.740	read_stream/2	1149
4.741	read_stream/3	1149
4.742	read_stream_by_line/2	1149
4.743	read_stream_by_line/3	1149
4.744	read_term_from_atom/3	1149
4.745	read_term_from_chars/3	1149
4.746	read_term_from_chars/4	1149
4.747	read_term_from_codes/3	1150
4.748	read_term_from_codes/4	1150
4.749	readme/1	1150
4.750	readme/2	1150
4.751	recorda/2	1150
4.752	recorda/3	1150
4.753	recorded/2	1150
4.754	recorded/3	1150
4.755	recordz/2	1151
4.756	recordz/3	1151
4.757	relative_standard_deviation/2	1151
4.758	removeDependent/1	1151

4.759	remove_duplicates/2	1151
4.760	rename_file/2	1151
4.761	replace/3	1151
4.762	replace_sub_atom/4	1151
4.763	rescale/3	1152
4.764	reset/0	1152
4.765	reset/1	1152
4.766	reset_counter/1	1152
4.767	reset_counters/0	1152
4.768	reset_flags/0	1152
4.769	reset_flags/1	1152
4.770	reset_genint/0	1153
4.771	reset_genint/1	1153
4.772	reset_gensym/0	1153
4.773	reset_gensym/1	1153
4.774	reset_monitor/0	1153
4.775	reset_seed/0	1153
4.776	restore/1	1153
4.777	restore/2	1153
4.778	reverse/2	1154
4.779	rewind/2	1154
4.780	rewind/3	1154
4.781	rlibraries/1	1154
4.782	rlibraries/2	1154
4.783	rlibrary/1	1154
4.784	rlibrary/2	1154
4.785	rlibrary_score/2	1155
4.786	rule/2	1155
4.787	rule/3	1155
4.788	rule/4	1155
4.789	run/0	1155
4.790	run/1	1155
4.791	run/2	1155
4.792	run_test_sets/1	1155
4.793	same_length/2	1156
4.794	same_length/3	1156
4.795	save/0	1156
4.796	save/1	1156
4.797	save/2	1156
4.798	scalar_product/3	1156
4.799	scan_left/4	1156
4.800	scan_left_1/3	1156
4.801	scan_right/4	1157
4.802	scan_right_1/3	1157
4.803	search/1	1157
4.804	select/3	1157
4.805	select/4	1157
4.806	selectchk/3	1157
4.807	selectchk/4	1157
4.808	send/3	1158
4.809	sequence/3	1158
4.810	sequence/4	1158
4.811	sequence/5	1158
4.812	sequential_occurrences/2	1158

4.813	sequential_occurrences/3	1158
4.814	serve/3	1158
4.815	set/1	1158
4.816	set/4	1159
4.817	set_element/2	1159
4.818	set_field/2	1159
4.819	set_flag_value/2	1159
4.820	set_flag_value/3	1159
4.821	set_monitor/4	1159
4.822	set_seed/1	1159
4.823	set_spy_point/4	1159
4.824	set_write_max_depth/1	1160
4.825	setup/0	1160
4.826	shell/1	1160
4.827	shell/2	1160
4.828	shell_command/1	1160
4.829	shrink/3	1160
4.830	shrink_sequence/3	1160
4.831	shrinker/1	1160
4.832	sign//1	1161
4.833	singletons/2	1161
4.834	size/2	1161
4.835	skewness/2	1161
4.836	sleep/1	1161
4.837	softmax/2	1161
4.838	softmax/3	1161
4.839	sort/2	1161
4.840	sort/3	1162
4.841	sort/4	1162
4.842	source_file_extension/1	1162
4.843	space//0	1162
4.844	spaces//0	1162
4.845	split/3	1162
4.846	split/4	1162
4.847	spy/1	1162
4.848	spy/3	1163
4.849	spy/4	1163
4.850	spy_point/4	1163
4.851	spying/1	1163
4.852	spying/3	1163
4.853	spying/4	1163
4.854	standard_cauchy/3	1163
4.855	standard_deviation/2	1163
4.856	standard_exponential/1	1164
4.857	standard_gamma/2	1164
4.858	standard_normal/1	1164
4.859	standard_t/2	1164
4.860	start/0	1164
4.861	start_redirect_to_file/2	1164
4.862	started_by/2	1164
4.863	starts/2	1164
4.864	stop/0	1165
4.865	stop_redirect_to_file/0	1165
4.866	stream_to_bytes/2	1165

4.867	stream_to_bytes/3	1165
4.868	stream_to_chars/2	1165
4.869	stream_to_chars/3	1165
4.870	stream_to_codes/2	1165
4.871	stream_to_codes/3	1165
4.872	stream_to_terms/2	1166
4.873	stream_to_terms/3	1166
4.874	subclass/1	1166
4.875	subclasses/1	1166
4.876	sublist/2	1166
4.877	subsequence/3	1166
4.878	subsequence/4	1166
4.879	subset/2	1166
4.880	substitute/4	1167
4.881	subsumes/2	1167
4.882	subterm/2	1167
4.883	subtract/3	1167
4.884	succ/2	1167
4.885	suffix/2	1167
4.886	suffix/3	1167
4.887	sum/2	1168
4.888	superclass/1	1168
4.889	superclasses/1	1168
4.890	suspend_monitor/0	1168
4.891	swap/2	1168
4.892	swap_consecutive/2	1168
4.893	syndiff/3	1168
4.894	tab/0	1168
4.895	tabs/0	1169
4.896	take/3	1169
4.897	temporary_directory/1	1169
4.898	term_expansion/2	1169
4.899	terms_to_array/2	1169
4.900	test/1	1169
4.901	time_stamp/1	1169
4.902	timeout/1	1169
4.903	timestamp/2	1170
4.904	timestamp/8	1170
4.905	today/3	1170
4.906	tolerance_equal/4	1170
4.907	top/3	1170
4.908	top_next/5	1170
4.909	trace/0	1170
4.910	trace_event/2	1170
4.911	transpose/2	1171
4.912	triangular/4	1171
4.913	true/1	1171
4.914	type/1	1171
4.915	unexpected/1	1171
4.916	unexpecteds/2	1171
4.917	uniform/1	1171
4.918	uniform/3	1171
4.919	uninstall/0	1172
4.920	uninstall/1	1172

4.921	uninstall/2	1172
4.922	union/3	1172
4.923	union/4	1172
4.924	union_all/3	1172
4.925	unpin/0	1172
4.926	unpin/1	1172
4.927	unzip/2	1173
4.928	update/0	1173
4.929	update/1	1173
4.930	update/2	1173
4.931	update/3	1173
4.932	update/4	1173
4.933	update/5	1173
4.934	update_in/4	1174
4.935	update_in/5	1174
4.936	uuid_null/1	1174
4.937	uuid_v1/2	1174
4.938	uuid_v4/1	1174
4.939	valid/1	1174
4.940	valid/2	1174
4.941	valid/3	1175
4.942	valid_date/3	1175
4.943	valid_option/1	1175
4.944	valid_options/1	1175
4.945	validate/1	1175
4.946	value/1	1175
4.947	value/3	1175
4.948	value_reference/2	1175
4.949	values/2	1176
4.950	variables/2	1176
4.951	variance/2	1176
4.952	variant/2	1176
4.953	varnumbers/2	1176
4.954	varnumbers/3	1176
4.955	verify_commands_availability/0	1176
4.956	version/6	1176
4.957	versions/3	1177
4.958	void/1	1177
4.959	void_element/1	1177
4.960	von_mises/3	1177
4.961	wald/3	1177
4.962	wall_time/1	1177
4.963	weibull/3	1177
4.964	weighted_mean/3	1177
4.965	welcome/0	1178
4.966	when/2	1178
4.967	whiledo/2	1178
4.968	white_space//0	1178
4.969	white_spaces//0	1178
4.970	with_output_to/2	1178
4.971	without//2	1178
4.972	working_directory/1	1178
4.973	write_file/3	1179
4.974	write_max_depth/1	1179

4.975	write_stream/3	1179
4.976	write_term_to_atom/3	1179
4.977	write_term_to_chars/3	1179
4.978	write_term_to_chars/4	1179
4.979	write_term_to_codes/3	1179
4.980	write_term_to_codes/4	1179
4.981	write_to_atom/2	1180
4.982	write_to_chars/2	1180
4.983	write_to_codes/2	1180
4.984	z_normalization/2	1180
4.985	zero_or_more/0	1180
4.986	zero_or_more/1	1180
4.987	zero_or_more/2	1180
4.988	zip/2	1180
4.989	zip/3	1181
4.990	zip_at_index/4	1181
5	Indices and tables	1183
	Index	1185

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 arbitrary

category

1.1.1 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: 2:35:1

Date: 2024-08-13

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: `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)`.
- 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`, `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](#)

1.2 assertions

object

1.2.1 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

1.2.2 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

1.2.3 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)

1.3 assignvars

object

1.3.1 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 ($=$)/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

1.3.2 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](#)

1.4 base64

object

1.4.1 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

1.4.2 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)

1.5 cbor

object

1.5.1 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

1.5.2 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)

1.6 code_metrics

object

1.6.1 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: 0:5:2

Date: 2024-05-15

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

1.6.2 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

1.6.3 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

1.6.4 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

1.6.5 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

1.6.6 `coupling_metric`

Computes entity efferent coupling, afferent coupling, and instability.

Availability:

`logtalk_load(code_metrics(loader))`

Author: Ebrahim Azarisooreh and Paulo Moura

Version: 0:14:0

Date: 2024-03-27

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.
- Entity score: Represented as the compound term `ce_ca_i_a(Ce,Ca,I,A)`.
- 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

1.6.7 dit__metric

Analyzes the depth of inheritance for objects, protocols, and categories.

Availability:

```
logtalk__load(code__metrics(loader))
```

Author: Ebrahim Azarisooreh

Version: 0:6:1

Date: 2024-03-28

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

1.6.8 doc_metric

Entity and entity predicates documentation score.

Availability:

```
logtalk_load(code_metrics(loader))
```

Author: Paulo Moura

Version: 0:13:0

Date: 2022-05-05

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

1.6.9 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

1.6.10 `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

1.6.11 `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: 0:14:1

Date: 2024-05-08

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

1.6.12 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: 0:5:1

Date: 2024-05-08

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

1.6.13 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: 0:7:1

Date: 2024-05-08

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

1.6.14 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: 0:6:2

Date: 2024-05-15

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)

1.7 core

category

1.7.1 core__messages

Logtalk core (compiler and runtime) default message tokenization.

Availability:

built_in

Author: Paulo Moura

Version: 1:145:0

Date: 2025-11-15

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

1.7.2 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

1.7.3 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`

1.7.4 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 `writeln/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 `writeln/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

1.7.5 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

1.7.6 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)

1.8 coroutining

object

1.8.1 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)

1.9 csv

object

1.9.1 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)`

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

1.9.2 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.

CSV file and stream reading and writing predicates.

Availability:

```
logtalk_load(csv(loader))
```

Author: Jacinto Dávila and Paulo Moura

Version: 2:1:0

Date: 2023-11-15

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

1.9.3 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

1.9.4 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)

1.10 dates

object

1.10.1 date

Date predicates.

Availability:

```
logtalk_load(dates(loader))
```

Author: Paulo Moura

Version: 1:2:0

Date: 2014-09-27

Compilation flags:

```
static, context_switching_calls
```

Implements:

```
public datep
```

Uses:

```
os
```

Remarks:

```
(none)
```

Inherited public predicates:

```
days_in_month/3 leap_year/1 name_of_day/3 name_of_month/3 today/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)

protocol

1.10.2 datep

Date protocol.

Availability:

logtalk__load(dates(loader))

Author: Paulo Moura

Version: 1:1:0

Date: 2005-03-17

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
- 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.

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`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

 See also

`date`, `timep`

object

1.10.3 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

1.10.4 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

1.11 dead_code_scanner

object

1.11.1 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: 0:15:2
Date: 2024-10-21

Compilation flags:
static, context_switching_calls

Imports:
public options

Uses:
list
logtalk
os
type

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.
- 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 due 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 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
- predicates/2
- predicate/2
- 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

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

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

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

category

1.11.2 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: 0:8:0

Date: 2024-05-07

Compilation flags:

```
static
```

Provides:

```
logtalk::message_prefix_stream/4
```

```
logtalk::message_tokens//2
```

Remarks:

(none)

Inherited public predicates:

(none)

- Public predicates
- Protected predicates
- Private predicates
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.12 debug_messages

object

1.12.1 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)

1.13 debugger

object

1.13.1 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: 7:12:2

Date: 2025-09-05

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

1.13.2 debugger_messages

Logtalk debugger tool default message translations.

Availability:

`logtalk_load(debugger(loader))`

Author: Paulo Moura

Version: 3:9:0

Date: 2025-09-03

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

1.13.3 debuggerp

Debugger protocol.

Availability:

`logtalk__load(debugger(loader))`

Author: Paulo Moura

Version: 3:6:0

Date: 2025-09-05

Compilation flags:

`static`

Dependencies:

(none)

Remarks:

- Debugger help: Type the character `h` (condensed help) or the character `?` (extended help) at a leased 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

1.13.4 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)

1.14 dependents

category

1.14.1 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

1.14.2 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](#)

1.15 diagrams

object

1.15.1 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: 1:3:0

Date: 2025-10-27

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

1.15.2 diagram(Format)

- Format - Graph language file format.

Common predicates for generating diagrams.

Availability:

```
logtalk_load(diagrams(loader))
```

Author: Paulo Moura

Version: 3:16:0

Date: 2025-10-27

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
- 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

Operators

(none)

object

1.15.3 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

1.15.4 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

1.15.5 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

1.15.6 `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: 3:0:1

Date: 2024-04-01

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

1.15.7 `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`

1.15.8 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

1.15.9 `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

<code>directory__dependency__diagram(Format),</code>	<code>file__dependency__diagram(Format),</code>	<code>li-</code>
<code>brary__dependency__diagram(Format)</code>		

object

1.15.10 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: 3:12:0

Date: 2025-10-27

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

1.15.11 `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: 2:0:0

Date: 2014-01-01

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, xref_diagram`

object

1.15.12 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: 2:60:0

Date: 2024-12-04

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public diagram(Format)
```

Uses:

```
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),	xref_diagram(Format),	li-
brary_diagram(Format)			

object

1.15.13 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

1.15.14 `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

1.15.15 `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

1.15.16 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

1.15.17 `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

<code>file_dependency_diagram(Format),</code>	<code>directory_dependency_diagram(Format),</code>	<code>li-</code>
<code>brary_dependency_diagram(Format)</code>		

protocol

1.15.18 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

1.15.19 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

1.15.20 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

1.15.21 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

1.15.22 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

1.15.23 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: 2:33:1

Date: 2024-04-01

Compilation flags:

static, context_switching_calls

Imports:

public library_diagram(Format)

Uses:

entity_diagram(Format)

list

logtalk

modules_diagram_support

Remarks:

(none)

Inherited public predicates:

all_files/0 all_files/1 all_libraries/0 all_libraries/1 check_option/1 check_options/1
default_option/1 default_options/1 diagram_description/1 diagram_name_suffix/1 directories/2
directories/3 directory/1 directory/2 directory/3 files/1 files/2 files/3 format_object/1
libraries/1 libraries/2 libraries/3 library/1 library/2 option/2 option/3 rdirectory/1
rdirectory/2 rdirectory/3 rlibrary/1 rlibrary/2 valid_option/1 valid_options/1

- Public predicates
- Protected predicates
- Private predicates
 - sub_diagram_/1
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

sub_diagram_/1

Table of library sub-diagrams to support their generation.

Compilation flags:

dynamic

Template:

sub_diagram_(Library)

Mode and number of proofs:

sub_diagram_(?atom) - zero_or_more

Operators

(none)

➡ See also

library_load_diagram(Format), directory_load_diagram(Format), file_load_diagram(Format), entity_diagram(Format)

category

1.15.24 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

1.15.25 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

1.15.26 `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

1.15.27 mermaid_graph_language

Predicates for generating graph files using Mermaid.

Availability:

`logtalk_load(diagrams(loader))`

Author: Paulo Moura

Version: 0:3:1

Date: 2025-11-11

Compilation flags:

`static, context_switching_calls`

Implements:

`public graph_language_protocol`

Imports:

`public options`

Provides:

`graph_language_registry::language_object/2`

Uses:

`list`

`os`

`term_io`

`user`

Remarks:

(none)

Inherited public predicates:

```
check_option/1 check_options/1 default_option/1 default_options/1 edge/6 file_footer/3
file_header/3 graph_footer/5 graph_header/5 node/7 option/2 option/3 output_file_name/2
valid_option/1 valid_options/1
```

- Public predicates
- Protected predicates
- Private predicates
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

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 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

1.15.29 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

1.15.30 `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

1.15.31 xref_diagram

Predicates for generating predicate call cross-referencing diagrams in DOT format.

Availability:

```
logtalk_load(diagrams(loader))
```

Author: Paulo Moura

Version: 2:0:0

Date: 2014-01-01

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

[entity__diagram](#), [inheritance__diagram](#), [uses__diagram](#)

object

1.15.32 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: 2:85:1

Date: 2025-10-27

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

entity_diagram(Format), inheritance_diagram(Format), uses_diagram(Format)

1.16 dictionaries

object

1.16.1 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: 1:4:1

Date: 2025-05-28

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
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

1.16.2 [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: 2:11:1

Date: 2022-05-05

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
 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](#)

protocol

1.16.3 dictionaryp

Dictionary protocol.

Availability:

```
logtalk_load(dictionaries(loader))
```

Author: Paulo Moura

Version: 2:4:0

Date: 2024-10-02

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/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/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](#)

object

1.16.4 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: 1:9:0

Date: 2021-04-12

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](#)
[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`, `bintree`

1.17 dif

object

1.17.1 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)

1.18 doclet

object

1.18.1 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)

1.19 edcg

object

1.19.1 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
```

1.20 events

object

1.20.1 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

1.20.2 [before_event_registry](#)

Before events registry predicates.

Availability:

`logtalk_load(events(loader))`

Author: Paulo Moura

Version: 1:1:0

Date: 2009-10-08

Compilation flags:

`static, context_switching_calls, events`

Implements:

`public event_registryp`

Remarks:

(none)

Inherited public predicates:

`del_monitors/0 del_monitors/4 monitor/1 monitor/4 monitored/1 monitors/1 set_monitor/4`

- [Public predicates](#)
- [Protected predicates](#)
- [Private predicates](#)
- [Operators](#)

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

➡ See also

`after_event_registry`, `monitorp`

object

1.20.3 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_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)

protocol

1.20.4 event_registryp

Event registry protocol.

Availability:

logtalk_load(events(loader))

Author: Paulo Moura

Version: 1:1:0

Date: 2009-10-08

Compilation flags:

static

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- Public predicates
 - monitors/1
 - monitor/1

- monitored/1
- monitor/4
- set_monitor/4
- del_monitors/4
- del_monitors/0
- Protected predicates
- Private predicates
- Operators

Public predicates

monitors/1

Returns a list of all current monitors.

Compilation flags:

static

Template:

monitors(Monitors)

Mode and number of proofs:

monitors(-list(object_identifier)) - one

monitor/1

Monitor is an object playing the role of a monitor.

Compilation flags:

static

Template:

monitor(Monitor)

Mode and number of proofs:

monitor(-object_identifier) - zero_or_more

monitor(+object_identifier) - zero_or_one

monitored/1

Returns a list of all currently monitored objects.

Compilation flags:

static

Template:

monitored(Objects)

Mode and number of proofs:

monitored(-list(object_identifier)) - one

monitor/4

True if the arguments describe a currently defined monitored event.

Compilation flags:

static

Template:

monitor(Object,Message,Sender,Monitor)

Mode and number of proofs:

monitor(?object_identifier,?nonvar,?object_identifier,?object_identifier) - zero_or_more

set_monitor/4

Sets a monitor for the set of matching events.

Compilation flags:

static

Template:

set_monitor(Object,Message,Sender,Monitor)

Mode and number of proofs:

set_monitor(?object_identifier,?nonvar,?object_identifier,+object_identifier) - zero_or_one

`del_monitors/4`

Deletes all matching monitored events.

Compilation flags:

`static`

Template:

`del_monitors(Object,Message,Sender,Monitor)`

Mode and number of proofs:

`del_monitors(?object_identifier,?nonvar,?object_identifier,?object_identifier) - one`

`del_monitors/0`

Deletes all monitored events.

Compilation flags:

`static`

Mode and number of proofs:

`del_monitors - one`

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

 See also

`event_registry`, `monitorp`

`category`

1.20.5 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

1.20.6 monitorp

Monitor protocol.

Availability:

logtalk_load(events(loader))

Author: Paulo Moura

Version: 1:0:0

Date: 2000-07-24

Compilation flags:

static

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- Public predicates
 - monitor_activated/0
 - activate_monitor/0
 - suspend_monitor/0
 - reset_monitor/0
 - spy_point/4
 - set_spy_point/4
 - del_spy_points/4
- Protected predicates
- Private predicates
- Operators

Public predicates

monitor_activated/0

True if monitor is currently active.

Compilation flags:

static

Mode and number of proofs:

monitor_activated - zero_or_one

`activate_monitor/0`

Activates all spy points and start monitoring.

Compilation flags:

`static`

Mode and number of proofs:

`activate_monitor - one`

`suspend_monitor/0`

Suspends monitoring, deactivating all spy points.

Compilation flags:

`static`

Mode and number of proofs:

`suspend_monitor - one`

`reset_monitor/0`

Resets monitor, deactivating and deleting all spy points.

Compilation flags:

`static`

Mode and number of proofs:

`reset_monitor - one`

spy_point/4

Current spy point.

Compilation flags:

static

Template:

spy_point(Event, Object, Message, Sender)

Mode and number of proofs:

spy_point(?event, ?object, ?callable, ?object) - zero_or_more

set_spy_point/4

Sets a spy point.

Compilation flags:

static

Template:

set_spy_point(Event, Object, Message, Sender)

Mode and number of proofs:

set_spy_point(?event, ?object, ?callable, ?object) - one

del_spy_points/4

Deletes all matching spy points.

Compilation flags:

static

Template:

del_spy_points(Event, Object, Message, Sender)

Mode and number of proofs:

del_spy_points(@event, @object, @callable, @object) - one

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

 See also

[monitor](#), [event_registryp](#)

1.21 `expand_library_alias_paths`

object

1.21.1 `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)

1.22 expecteds

object

1.22.1 either

Types and predicates for extended type-checking and handling of expected terms.

Availability:

`logtalk_load(expecteds(loader))`

Author: Paulo Moura

Version: 0:7:0

Date: 2021-01-03

Compilation flags:

static, context_switching_calls

Provides:

type::type/1
type::check/2
arbitrary::arbitrary/1
arbitrary::arbitrary/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` predicates to allow generating random values for the `either(ValueType, ErrorType)` type.

Inherited public predicates:

(none)

- Public predicates
 - `expecteds/2`
 - `unexpecteds/2`
 - `partition/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

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

1.22.2 `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: 2:1:0

Date: 2021-01-03

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
- 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`

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

1.22.3 `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: 1:5:0

Date: 2020-01-06

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`
- 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
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

 See also

[expected](#)

1.23 fcube

object

1.23.1 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

1.24 flags

category

1.24.1 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

1.24.2 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)

1.25 format

object

1.25.1 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)

1.26 genint

object

1.26.1 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

1.26.2 `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)

1.27 gensym

object

1.27.1 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

1.27.2 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)

1.28 git

object

1.28.1 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

1.28.2 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)

1.29 grammars

object

1.29.1 `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

1.29.2 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

1.29.3 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

1.29.4 `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)

1.30 heaps

object

1.30.1 heap(Order)

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: 1:1:0

Date: 2019-05-18

Compilation flags:

static, context__switching__calls

Implements:

public `heapp`

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

minheap, maxheap

protocol

1.30.2 heapp

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\(Ord\)](#)

object

1.30.3 maxheap

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 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)

object

1.30.4 minheap

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 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)

1.31 help

object

1.31.1 help

Command-line help for Logtalk libraries, entities, plus built-in control constructs, predicates, non-terminals, and methods.

Availability:

```
logtalk_load(help(loader))
```

Author: Paulo Moura

Version: 0:38:0

Date: 2025-10-06

Compilation flags:

```
static, context_switching_calls, complements(allow)
```

Implements:

public forwarding

Uses:

atom

os

user

Remarks:

(none)

Inherited public predicates:

apis/0 apis/1 forward/1 handbook/0 handbook/1 man/1

- Public predicates
 - help/0
 - (/)/2
 - (//)/2
 - completion/2
 - completions/2
 - built_in_directive/4
 - built_in_predicate/4
 - built_in_method/4
 - control_construct/4
 - built_in_non_terminal/4
 - library/0
 - library/1
 - entity/1
 - manuals/0
- Protected predicates
- Private predicates
- Operators

Public predicates

`help/0`

Prints instructions on how to use the help tool.

Compilation flags:

`static`

Mode and number of proofs:

`help - one`

`(/)/2`

Provides help on the Functor/Arity built-in control construct, directive, predicate, or method.

Compilation flags:

`static`

Template:

`Functor/Arity`

Mode and number of proofs:

`+atom/ +integer - zero__or__one`

`(//)/2`

Provides help on the Functor//Arity built-in non-terminal.

Compilation flags:

`static`

Template:

`Functor//Arity`

Mode and number of proofs:

`+atom// +integer - zero__or__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(Functor,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(Functor,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(Functor,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(Functor,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(Functor,Arity,Directory,Basename)`

Mode and number of proofs:

`built_in_non_terminal(?atom,?integer,-atom,-atom) - zero_or_more`

`library/0`

Provides help on the standard Logtalk library.

Compilation flags:

`static`

Mode and number of proofs:

`library - one`

`library/1`

Provides help on the standard Logtalk libraries, library predicates, and library non-terminals.

Compilation flags:

`static`

Template:

`library(Topic)`

Mode and number of proofs:

`library(+atom) - zero_or_one`

`library(+predicate_indicator) - zero_or_one`

`library(+non_terminal_indicator) - zero_or_one`

entity/1

Provides help on Logtalk entities (objects, protocols, or categories).

Compilation flags:

static

Template:

entity(Entity)

Mode and number of proofs:

entity(+entity_identifier) - zero_or_one

manuals/0

Provides access to the Logtalk User and Reference manuals.

Compilation flags:

static

Mode and number of proofs:

manuals - one

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

category

1.31.2 help_info_support

Experimental help predicates for inline browsing of the Texinfo versions of the Handbook and APIs documentation. Currently requires Ciao Prolog, ECLiPSe, GNU Prolog, XVM, SICStus Prolog, SWI-Prolog, Trealla Prolog, XSB, or YAP as the backend running on a POSIX system.

Availability:

```
logtalk_load(help(loader))
```

Author: Paulo Moura

Version: 0:9:0

Date: 2025-05-02

Compilation flags:

```
static
```

Complements:

```
help
```

Uses:

```
os
```

```
user
```

Remarks:

```
(none)
```

Inherited public predicates:

```
(none)
```

- Public predicates
 - handbook/0
 - handbook/1
 - apis/0
 - apis/1
 - man/1
- Protected predicates
- Private predicates
- Operators

Public predicates

[handbook/0](#)

Opens inline the Texinfo version of the Handbook.

Compilation flags:
static

Mode and number of proofs:
handbook - one

[handbook/1](#)

Opens inline the Texinfo version of the Handbook at the given topic.

Compilation flags:
static

Template:
handbook(Topic)
Mode and number of proofs:
handbook(+atom) - one
handbook(+predicate_indicator) - one
handbook(+non_terminal_indicator) - one

[apis/0](#)

Opens inline the Texinfo version of the APIs documentation.

Compilation flags:
static

Mode and number of proofs:
apis - one

[apis/1](#)

Opens inline the Texinfo version of the APIs documentation at the given topic.

Compilation flags:

static

Template:

apis(Topic)

Mode and number of proofs:

apis(+atom) - one

apis(+predicate_indicator) - one

apis(+non_terminal_indicator) - one

[man/1](#)

Opens inline the man page of the given script.

Compilation flags:

static

Template:

man(Script)

Mode and number of proofs:

man(+atom) - one

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.32 hierarchies

category

1.32.1 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 ancestors/1 class/1 classes/1 descendant/1 descendant_class/1 descendant_classes/1
descendant_instance/1 descendant_instances/1 descendants/1 instance/1 instances/1 leaf/1
leaf_class/1 leaf_classes/1 leaf_instance/1 leaf_instances/1 leaves/1 subclass/1 subclasses/1
superclass/1 superclasses/1`

- Public predicates
- Protected predicates
- Private predicates
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

protocol

1.32.2 `class_hierarchy`

Class hierarchy protocol.

Availability:

`logtalk_load(hierarchies(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2000-07-24

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
 - 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

1.32.3 `hierarchyp`

Common hierarchy protocol for prototype and class hierarchies.

Availability:

`logtalk_load(hierarchies(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2000-07-24

Compilation flags:

`static`

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- Public predicates
 - ancestor/1
 - ancestors/1
 - leaf/1
 - leaves/1
 - descendant/1
 - descendants/1
- Protected predicates
- Private predicates
- Operators

Public predicates

ancestor/1

Returns, by backtracking, all object ancestors.

Compilation flags:

static

Template:

ancestor(Ancestor)

Mode and number of proofs:

ancestor(?object) - zero_or_more

ancestors/1

List of all object ancestors.

Compilation flags:

static

Template:

ancestors(Ancestors)

Mode and number of proofs:

ancestors(-list) - one

`leaf/1`

Returns, by backtracking, all object leaves.

Compilation flags:

`static`

Template:

`leaf(Leaf)`

Mode and number of proofs:

`leaf(?object) - zero_or_more`

`leaves/1`

List of all object leaves.

Compilation flags:

`static`

Template:

`leaves(Leaves)`

Mode and number of proofs:

`leaves(-list) - one`

`descendant/1`

Returns, by backtracking, all object descendants.

Compilation flags:

`static`

Template:

`descendant(Descendant)`

Mode and number of proofs:

descendant(?object) - zero_or_more

descendants/1

List of all object descendants.

Compilation flags:

static

Template:

descendants(Descendants)

Mode and number of proofs:

descendants(-list) - one

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

category

1.32.4 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

1.32.5 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`

1.33 hook_flows

object

1.33.1 `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

1.33.2 `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)`

1.34 hook_objects

object

1.34.1 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

1.34.2 `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

1.34.3 `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

1.34.4 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

1.34.5 `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

1.34.6 `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

1.34.7 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

1.34.8 `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

1.34.9 `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

1.34.10 `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

1.34.11 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

1.34.12 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

1.34.13 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

1.34.14 `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`

1.35 html

category

1.35.1 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

1.35.2 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

1.35.3 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)

1.36 ids

object

1.36.1 ids

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

Availability:

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

Author: Paulo Moura

Version: 1:1:0

Date: 2025-06-03

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), uuid, ulid

object

1.36.2 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: 1:0:0

Date: 2022-11-23

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`, `uuid`, `ulid`

1.37 intervals

object

1.37.1 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

1.37.2 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](#)

1.38 iso8601

object

1.38.1 iso8601

ISO 8601 (and European civil calendar) compliant library of date predicates.

Availability:

```
logtalk_load(iso8601(loader))
```

Author: Daniel L. Dudley

Version: 1:0:3

Date: 2019-10-09

Compilation flags:

```
static, context_switching_calls
```

Uses:

[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`
 - `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]

```

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)

1.39 issue_creator

object

1.39.1 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)

1.40 java

object

1.40.1 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

1.40.2 `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

1.40.3 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

1.40.4 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

1.40.5 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

1.40.6 `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)

1.41 json

object

1.41.1 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

1.41.2 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

1.41.3 `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: 0:13:0

Date: 2024-07-16

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

1.41.4 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)

1.42 json_lines

object

1.42.1 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

1.42.2 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

1.42.3 `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: 1:0:0

Date: 2025-05-27

Compilation flags:

static, context_switching_calls

Implements:

public json_lines_protocol

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

1.42.4 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)

1.43 lgtdoc

object

1.43.1 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: 11:2:1

Date: 2025-10-07

Compilation flags:

`static, context_switching_calls`

Implements:

`public lgtdocp`

Imports:

`public options`

Uses:

`date`

`list`

`logtalk`

`os`

`type`

`user`

`varlist`

Remarks:

(none)

Inherited public predicates:

all/0 all/1 check_option/1 check_options/1 default_option/1 default_options/1 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
- 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`

Operators

(none)

category

1.43.2 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

1.43.3 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.xsl')])
yes
```

`directory/1`

Creates XML documenting files for all entities in a directory using default options.

Compilation flags:

static

Template:

directory(Directory)

Mode and number of proofs:

directory(+atom) - one

`files/2`

Creates XML documenting files for all entities in loaded source files using the specified options. The files can be given by name, basename, full path, or using library notation.

Compilation flags:

static

Template:

files(Files,Options)

Mode and number of proofs:

files(+list(atom),+list) - one

files/1

Creates XML documenting files for all entities in loaded source files using default options. The files can be given by name, basename, full path, or using library notation.

Compilation flags:

static

Template:

files(Files)

Mode and number of proofs:

files(+list(atom)) - one

file/2

Creates XML documenting files for all entities in a loaded source file using the specified options. The file can be given by name, basename, full path, or using library notation.

Compilation flags:

static

Template:

file(File,Options)

Mode and number of proofs:

file(+atom,+list) - one

file/1

Creates XML documenting files for all entities in a loaded source file using default options. The file can be given by name, basename, full path, or using library notation.

Compilation flags:

static

Template:

file(File)

Mode and number of proofs:

file(+atom) - one

all/1

Creates XML documenting files for all loaded entities using the specified options.

Compilation flags:

static

Template:

all(Options)

Mode and number of proofs:

all(+list) - one

all/0

Creates XML documenting files for all loaded entities using default options.

Compilation flags:

static

Mode and number of proofs:

all - one

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

 See also

lgtdoc

1.44 lgtunit

object

1.44.1 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: 6:0:0

Date: 2025-04-07

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

1.44.2 `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: 3:2:0

Date: 2023-04-11

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(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

1.44.3 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: 22:3:0
Date: 2025-10-20

Compilation flags:
 static, context_switching_calls

Implements:
 public expanding
Imports:
 public lgtunit_messages
Provides:

logtalk::trace_event/2

Uses:

fast_random
list
logtalk
os
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:

failed_test_reason//1 goal_expansion/2 term_expansion/2

- 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
 - 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
 - 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`

`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 an empty 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

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`

1.44.4 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

1.44.5 `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

1.44.6 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

1.44.7 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

1.44.8 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

1.44.9 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

1.44.10 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

1.44.11 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)

1.45 library

protocol

1.45.1 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`

1.45.2 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

1.45.3 streamvars

Stream variables (supporting logical, backtracable, adding and retrieving of terms).

Availability:

`logtalk_load(library(loader))`

Author: Nobukuni Kino and Paulo Moura

Version: 1:3:0

Date: 2019-06-15

Compilation flags:

`static, context__switching__calls`

Dependencies:

`(none)`

Remarks:

`(none)`

Inherited public predicates:

`(none)`

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

Public predicates

new/1

Makes Variable a stream variable. Initial state will be empty.

Compilation flags:

static

Template:

new(Variable)

Mode and number of proofs:

new(--streamvar) - one

Exceptions:

Variable is not a variable:

type__error(variable,Variable)

new/2

Makes Variable a stream variable and sets its initial state to Value.

Compilation flags:

static

Template:

new(Variable,Value)

Mode and number of proofs:

new(--streamvar,@nonvar) - one

Exceptions:

Variable is not a variable:

type__error(variable,Variable)

$(\leq)/2$

Sets the state of the stream variable *Variable* to *Value* (initializing the variable if needed).

Compilation flags:

static

Template:

Variable \leq *Value*

Mode and number of proofs:

$(?streamvar)\leq(@nonvar) - one$

$(\Rightarrow)/2$

Unifies *Value* with the current state of the stream variable *Variable*.

Compilation flags:

static

Template:

Variable \Rightarrow *Value*

Mode and number of proofs:

$+streamvar\Rightarrow ?nonvar - zero_or_one$

Protected predicates

(none)

Private predicates

(none)

Operators

`op(100,xfx,<=)`

Scope:
public

`op(100,xfx,>=)`

Scope:
public

1.46 listing

category

1.46.1 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)

1.47 logging

object

1.47.1 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

1.47.2 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

1.47.3 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

1.48 loops

object

1.48.1 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

1.48.2 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

1.49 meta

object

1.49.1 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

1.49.2 metap

Useful meta-predicates protocol.

Availability:

logtalk_load(meta(loader))

Author: Paulo Moura

Version: 6:1:0

Date: 2015-12-23

Compilation flags:

static

Dependencies:

(none)

Remarks:

(none)

Inherited public predicates:

(none)

- Public predicates
 - include/3
 - exclude/3
 - findall__member/4
 - findall__member/5
 - partition/4
 - partition/6
 - fold__left/4
 - fold__left__1/3
 - scan__left/4
 - scan__left__1/3
 - fold__right/4
 - fold__right__1/3
 - scan__right/4
 - scan__right__1/3
 - map/2
 - map/3
 - map/4
 - map/5
 - map/6
 - map/7
 - map/8
 - map_reduce/5

- Protected predicates
- Private predicates
- Operators

Public predicates

`include/3`

Returns a list of all list elements that satisfy a predicate.

Compilation flags:

`static`

Template:

`include(Closure,List,Included)`

Meta-predicate template:

`include(1,*,*)`

Mode and number of proofs:

`include(+callable,+list,-list) - one`

`exclude/3`

Returns a list of all list elements that fail to satisfy a predicate.

Compilation flags:

`static`

Template:

`exclude(Closure,List,Excluded)`

Meta-predicate template:

`exclude(1,*,*)`

Mode and number of proofs:

`exclude(+callable,+list,-list) - one`

[findall_member/4](#)

Finds all members of a list that satisfy a given test.

Compilation flags:

static

Template:

findall_member(Member,List,Test,Result)

Meta-predicate template:

findall_member(*,*,0,*)

Mode and number of proofs:

findall_member(@term,+list,@callable,-list) - one

[findall_member/5](#)

Finds all members of a list that satisfy a given test appending the given tail to the result.

Compilation flags:

static

Template:

findall_member(Member,List,Test,Result,Tail)

Meta-predicate template:

findall_member(*,*,0,*,*)

Mode and number of proofs:

findall_member(@term,+list,@callable,-list,+list) - one

[partition/4](#)

Partition a list of elements in two lists using a predicate.

Compilation flags:

static

Template:

partition(Closure,List,Included,Excluded)

Meta-predicate template:

```
partition(1,*,*,*)
```

Mode and number of proofs:

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

[partition/6](#)

Partitions a list in lists with values less, equal, and greater than a given value using a comparison predicate with the same argument order as [compare/3](#).

Compilation flags:

```
static
```

Template:

```
partition(Closure,List,Value,Less,Equal,Greater)
```

Meta-predicate template:

```
partition(3,*,*,*,*,*)
```

Mode and number of proofs:

```
partition(+callable,+list,@term,-list,-list,-list) - one
```

[fold_left/4](#)

List folding (left associative). Closure is extended with three arguments: accumulator, list element, and updated accumulator.

Compilation flags:

```
static
```

Template:

```
fold_left(Closure,Accumulator,List,Result)
```

Meta-predicate template:

```
fold_left(3,*,*,*)
```

Mode and number of proofs:

```
fold_left(+callable,?term,+list,?term) - zero_or_more
```

fold_left_1/3

List folding (left associative). Closure is extended with three arguments: accumulator, list element, and updated accumulator. The initial value of the accumulator is the list first element. Fails for empty lists.

Compilation flags:

static

Template:

fold_left_1(Closure,List,Result)

Meta-predicate template:

fold_left_1(3,*,*)

Mode and number of proofs:

fold_left_1(+callable,+list,?term) - zero_or_more

scan_left/4

List scanning (left associative). Closure is extended with three arguments: accumulator, list element, and updated accumulator.

Compilation flags:

static

Template:

scan_left(Closure,Accumulator,List,Results)

Meta-predicate template:

scan_left(3,*,*,*)

Mode and number of proofs:

scan_left(+callable,?term,+list,?list) - zero_or_more

scan_left_1/3

List scanning (left associative). Closure is extended with three arguments: accumulator, list element, and updated accumulator. The accumulator is initialized with the list first element. Fails for empty lists.

Compilation flags:

static

Template:

```
scan_left_1(Closure,List,Results)
```

Meta-predicate template:

```
scan_left_1(3,*,*)
```

Mode and number of proofs:

```
scan_left_1(+callable,+list,?list) - zero_or_more
```

`fold_right/4`

List folding (right associative). Closure is extended with three arguments: list element, accumulator, and updated accumulator.

Compilation flags:

```
static
```

Template:

```
fold_right(Closure,Accumulator,List,Result)
```

Meta-predicate template:

```
fold_right(3,*,*,*)
```

Mode and number of proofs:

```
fold_right(+callable,?term,+list,?term) - zero_or_more
```

`fold_right_1/3`

List folding (right associative). Closure is extended with three arguments: list element, accumulator, and updated accumulator. The initial value of the accumulator is the list first element. Fails for empty lists.

Compilation flags:

```
static
```

Template:

```
fold_right_1(Closure,List,Result)
```

Meta-predicate template:

```
fold_right_1(3,*,*)
```

Mode and number of proofs:

```
fold_right_1(+callable,+list,?term) - zero_or_more
```

`scan_right/4`

List scanning (right associative). Closure is extended with three arguments: list element, accumulator, and updated accumulator.

Compilation flags:

`static`

Template:

`scan_right(Closure,Accumulator,List,Results)`

Meta-predicate template:

`scan_right(3,*,*,*)`

Mode and number of proofs:

`scan_right(+callable,?term,+list,?list) - zero_or_more`

`scan_right_1/3`

List scanning (right associative). Closure is extended with three arguments: list element, accumulator, and updated accumulator. The accumulator is initialized with the list first element. Fails for empty lists.

Compilation flags:

`static`

Template:

`scan_right_1(Closure,List,Results)`

Meta-predicate template:

`scan_right_1(3,*,*)`

Mode and number of proofs:

`scan_right_1(+callable,+list,?list) - zero_or_more`

`map/2`

True if the predicate succeeds for each list element.

Compilation flags:

`static`

Template:

`map(Closure,List)`

Meta-predicate template:

`map(1,*)`

Mode and number of proofs:

`map(+callable,?list) - zero_or_more`

`map/3`

List mapping predicate taken arguments from two lists of elements.

Compilation flags:

`static`

Template:

`map(Closure,List1,List2)`

Meta-predicate template:

`map(2,*,*)`

Mode and number of proofs:

`map(+callable,?list,?list) - zero_or_more`

`map/4`

List mapping predicate taken arguments from three lists of elements.

Compilation flags:

`static`

Template:

`map(Closure,List1,List2,List3)`

Meta-predicate template:

`map(3,*,*,*)`

Mode and number of proofs:

`map(+callable,?list,?list,?list) - zero_or_more`

map/5

List mapping predicate taken arguments from four lists of elements.

Compilation flags:

static

Template:

map(Closure,List1,List2,List3,List4)

Meta-predicate template:

map(4,*,*,*,*)

Mode and number of proofs:

map(+callable,?list,?list,?list,?list) - zero_or_more

map/6

List mapping predicate taken arguments from five lists of elements.

Compilation flags:

static

Template:

map(Closure,List1,List2,List3,List4,List5)

Meta-predicate template:

map(5,*,*,*,*,*)

Mode and number of proofs:

map(+callable,?list,?list,?list,?list,?list) - zero_or_more

map/7

List mapping predicate taken arguments from six lists of elements.

Compilation flags:

static

Template:

map(Closure,List1,List2,List3,List4,List5,List6)

Meta-predicate template:

```
map(6,*,*,*,*,*,*)
```

Mode and number of proofs:

```
map(+callable,?list,?list,?list,?list,?list,?list) - zero_or_more
```

map/8

List mapping predicate taken arguments from seven lists of elements.

Compilation flags:

```
static
```

Template:

```
map(Closure,List1,List2,List3,List4,List5,List6,List7)
```

Meta-predicate template:

```
map(7,*,*,*,*,*,*,*)
```

Mode and number of proofs:

```
map(+callable,?list,?list,?list,?list,?list,?list,?list) - zero_or_more
```

map_reduce/5

Map a list and apply a fold left (reduce) to the resulting list.

Compilation flags:

```
static
```

Template:

```
map_reduce(Map,Reduce,Accumulator,List,Result)
```

Meta-predicate template:

```
map_reduce(2,3,*,*,*)
```

Mode and number of proofs:

```
map_reduce(+callable,+callable,+term,?list,?term) - zero_or_more
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

 See also

[meta](#)

1.50 meta_compiler

object

1.50.1 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](#)

1.51 metagol

object

1.51.1 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

1.51.2 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)

1.52 mutations

object

1.52.1 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

1.52.2 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

1.52.3 `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

1.52.4 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`

1.52.5 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`

1.52.6 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

1.52.7 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

1.53 nested_dictionaries

object

1.53.1 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

1.53.2 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

1.53.3 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-brackted term representation from a (nested) dictionary.

Compilation flags:

`static`

Template:

`as_curly_bracketed(Dictionary,Term)`

Mode and number of proofs:

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

`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

1.53.4 `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](#)

1.54 optionals

object

1.54.1 maybe

Types and predicates for type-checking and handling optional terms. Inspired by Haskell.

Availability:

`logtalk_load(optionals(loader))`

Author: Paulo Moura

Version: 0:7:0

Date: 2021-01-03

Compilation flags:

`static, context_switching_calls`

Provides:

`type::type/1`

`type::check/2`

`arbitrary::arbitrary/1`

`arbitrary::arbitrary/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` predicates to allow generating random values for the `maybe(Type)` type.

Inherited public predicates:

(none)

- Public predicates
 - `cat/2`
- 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`

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

1.54.2 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: 2:1:0

Date: 2021-01-03

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
- 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 or throws an error.

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 or throws an error.

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 error or failure, returns an empty optional.

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 error or failure, returns an empty optional.

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`

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

1.54.3 `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: 1:7:0

Date: 2019-11-26

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
- 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
```

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

 See also

[optional](#)

1.55 options

category

1.55.1 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

1.55.2 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](#)

1.56 os

object

1.56.1 `os`

Portable operating-system access predicates.

Availability:

```
logtalk_load(os(loader))
```

Author: Paulo Moura

Version: 1:102:0

Date: 2025-11-15

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

1.56.2 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

1.56.3 osp

Portable operating-system access protocol.

Availability:

`logtalk_load(os(loader))`

Author: Paulo Moura

Version: 1:41:0

Date: 2025-05-19

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.

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, TMP, TEMP, and TEMPDIR 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`

1.57 packs

protocol

1.57.1 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

1.57.2 packs

Pack handling predicates.

Availability:

logtalk__load(packs(loader))

Author: Paulo Moura

Version: 0:87:1

Date: 2025-08-20

Compilation flags:

static, context_switching_calls

Imports:

public packs__common

public options

Uses:

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
- search/1
- 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)

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)

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. Registries without installed packs are saved when using the option `save(all)` and skipped when using the option `save(installed)` (default).

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)

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:

- 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

1.57.3 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

1.57.4 packs_messages

Packs default message translations.

Availability:

logtalk_load(packs(loader))

Author: Paulo Moura

Version: 0:41:0

Date: 2025-05-23

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

1.57.5 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

1.57.6 registries

Registry handling predicates.

Availability:

`logtalk_load(packs(loader))`

Author: Paulo Moura

Version: 0:62:0

Date: 2025-08-20

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.
- `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:

- `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

1.57.7 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

1.57.8 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)

1.58 pddl_parser

object

1.58.1 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

1.58.2 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)

1.59 ports_profiler

object

1.59.1 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)

1.60 queues

object

1.60.1 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

1.60.2 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](#)

1.61 random

object

1.61.1 backend_random

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

Availability:

`logtalk_load(random(loader))`

Author: Paulo Moura

Version: 1:21:1

Date: 2025-04-07

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

1.61.2 [fast_random](#)

Fast 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: 2:12:1

Date: 2025-04-07

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.

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_/3
- 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_/3

Stores the current random generator seed values.

Compilation flags:

dynamic

Template:

seed__(S0,S1,S2)

Mode and number of proofs:

seed__(-integer,-integer,-integer) - one

Operators

(none)

➡ See also

[random](#), [backend_random](#)

protocol

1.61.3 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

1.61.4 `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: 2:12:1

Date: 2025-04-07

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.

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_/3`
- 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_/3

Stores the current random generator seed values.

Compilation flags:

dynamic

Template:

seed__(S0,S1,S2)

Mode and number of proofs:

seed__(-integer,-integer,-integer) - one

Operators

(none)

➡ See also

`fast_random`, `backend_random`

protocol

1.61.5 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

1.61.6 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`

1.62 reader

object

1.62.1 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)

1.63 recorded_database

object

1.63.1 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

1.63.2 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 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)

1.64 redis

object

1.64.1 redis

Redis client. Inspired by Sean Charles GNU Prolog Redis client.

Availability:

```
logtalk_load(redis(loader))
```

Author: Paulo Moura

Version: 0:5:1

Date: 2021-12-06

Compilation flags:

```
static, context_switching_calls
```

Provides:

```
logtalk::message_tokens//2
```

Uses:

```
list
```

```
logtalk
```

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.

Inherited public predicates:

(none)

- Public predicates
 - connect/1
 - connect/3
 - disconnect/1
 - send/3
 - console/1
- 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

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

1.65 sets

object

1.65.1 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)

object

1.65.2 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)

protocol

1.65.3 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)`

1.66 statistics

object

1.66.1 population

Statistical population represented as a list of numbers.

Availability:

```
logtalk_load(statistics(loader))
```

Author: Paulo Moura

Version: 1:3:0

Date: 2020-02-02

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public statistics
```

Remarks:

```
(none)
```

Inherited public predicates:

```
arithmetic_mean/2 average_deviation/3 coefficient_of_variation/2 fractile/3 geometric_mean/2  
harmonic_mean/2 kurtosis/2 max/2 mean_deviation/2 median/2 median_deviation/2 min/2  
min_max/3 modes/2 product/2 range/2 relative_standard_deviation/2 skewness/2  
standard_deviation/2 sum/2 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

1.66.2 sample

Statistical sample represented as a list of numbers.

Availability:

```
logtalk_load(statistics(loader))
```

Author: Paulo Moura

Version: 1:4:0

Date: 2020-02-02

Compilation flags:

```
static, context_switching_calls
```

Imports:

```
public statistics
```

Remarks:

(none)

Inherited public predicates:

```
arithmetic_mean/2 average_deviation/3 coefficient_of_variation/2 fractile/3 geometric_mean/2
harmonic_mean/2 kurtosis/2 max/2 mean_deviation/2 median/2 median_deviation/2 min/2
min_max/3 modes/2 product/2 range/2 relative_standard_deviation/2 skewness/2
standard_deviation/2 sum/2 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

1.66.3 statistics

Statistical calculations over a list of numbers.

Availability:

```
logtalk_load(statistics(loader))
```

Author: Paulo Moura

Version: 1:7:1

Date: 2023-05-29

Compilation flags:

```
static
```

Implements:

public statisticsp

Uses:

list

numberlist

Remarks:

(none)

Inherited public predicates:

arithmetic_mean/2 average_deviation/3 coefficient_of_variation/2 fractile/3 geometric_mean/2
 harmonic_mean/2 kurtosis/2 max/2 mean_deviation/2 median/2 median_deviation/2 min/2
 min_max/3 modes/2 product/2 range/2 relative_standard_deviation/2 skewness/2
 standard_deviation/2 sum/2 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
- 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
```

Operators

(none)

protocol

1.66.4 statisticsp

Statistical calculations over a list of numbers protocol.

Availability:

```
logtalk_load(statistics(loader))
```

Author: Paulo Moura

Version: 1:3:0

Date: 2022-06-20

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
 - 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. Wights are assume 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

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

1.67 term_io

object

1.67.1 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

1.67.2 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)

1.68 timeout

object

1.68.1 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)

1.69 toychr

object

1.69.1 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: 0:7:1

Date: 2024-03-15

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)

1.70 tsv

object

1.70.1 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)`

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

1.70.2 tsv(Header)

- Header - Header handling option with possible values skip and keep (default).

TSV file and stream reading and writing predicates.

Availability:

```
logtalk_load(tsv(loader))
```

Author: Paulo Moura

Version: 1:0:1

Date: 2024-03-11

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

1.70.3 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)

1.71 tutor

object

1.71.1 tutor

This object adds explanations and suggestions to selected compiler warning and error messages.

Availability:

`logtalk__load(tutor(loader))`

Author: Paulo Moura

Version: 0:84:0

Date: 2025-10-20

Compilation flags:

`static, context_switching_calls`

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:

(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)

1.72 types

object

1.72.1 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

1.72.2 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

1.72.3 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

1.72.4 [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

1.72.5 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

1.72.6 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

1.72.7 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

1.72.8 difflist

Difference list 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 [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 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

1.72.9 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

1.72.10 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

1.72.11 list

List predicates.

Availability:

logtalk_load(types(loader))

Author: Paulo Moura

Version: 4:3:0

Date: 2024-05-24

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 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

1.72.12 `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` `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

1.72.13 listp

List protocol.

Availability:

`logtalk_load(types(loader))`

Author: Paulo Moura

Version: 1:18:0

Date: 2024-05-24

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
 - 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`

`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

1.72.14 `natural`

Natural numbers data type predicates.

Availability:

`logtalk_load(types(loader))`

Author: Paulo Moura

Version: 2:0:0

Date: 2025-01-16

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
- Protected predicates
- Private predicates
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

object

1.72.15 number

Number data type predicates.

Availability:

`logtalk_load(types(loader))`

Author: Paulo Moura

Version: 1:14:0

Date: 2023-12-07

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 $100 * \text{epsilon}$ 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

1.72.16 numberlist

List of numbers predicates.

Availability:

logtalk_load(types(loader))

Author: Paulo Moura

Version: 1:16:0

Date: 2025-03-13

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 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 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

1.72.17 numberlistp

List of numbers protocol.

Availability:

`logtalk_load(types(loader))`

Author: Paulo Moura

Version: 1:10:0

Date: 2025-03-13

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
- 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.

Protected predicates

(none)

Private predicates

(none)

Operators

(none)

 See also

numberlist, listp, varlistp

object

1.72.18 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

1.72.19 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

1.72.20 term_p

Term utility predicates protocol.

Availability:

`logtalk_load(types(loader))`

Author: Paulo Moura

Version: 1:35:0

Date: 2022-05-13

Compilation flags:

`static`

Extends:

`public comparingp`

Remarks:

(none)

Inherited public predicates:

`(<)/2` `(=:)/2` `(=<)/2` `(=\=)/2` `(>)/2` `(>=)/2`

- Public predicates
 - `depth/2`
 - `ground/1`
 - `new/1`
 - `occurs/2`
 - `subsumes/2`

- subterm/2
- valid/1
- check/1
- variant/2
- variables/2
- singletons/2
- numbervars/3
- numbervars/1
- varnumbers/3
- varnumbers/2
- Protected predicates
- Private predicates
- Operators

Public predicates

depth/2

True if the depth of Term is Depth. The depth of atomic terms is zero; the depth of a compound term is one plus the maximum depth of its sub-terms.

Compilation flags:

static

Template:

depth(Term,Depth)

Mode and number of proofs:

depth(@term,?integer) - zero_or_one

ground/1

True if the argument is ground. Deprecated. Use the ground/1 standard predicate instead.

Compilation flags:

static

Template:

ground(Term)

Mode and number of proofs:

ground(@term) - zero_or_one

new/1

Creates a new term instance (if meaningful).

Compilation flags:

static

Template:

new(Term)

Mode and number of proofs:

new(-nonvar) - zero_or_one

occurs/2

True if the variable occurs in the term.

Compilation flags:

static

Template:

occurs(Variable,Term)

Mode and number of proofs:

occurs(@var,@term) - zero_or_one

subsumes/2

The first term subsumes the second term. Deprecated. Use the subsumes_term/2 standard predicate instead.

Compilation flags:

static

Template:

subsumes(General,Specific)

Mode and number of proofs:

subsumes(@term,@term) - zero_or_one

subterm/2

The first term is a subterm of the second term.

Compilation flags:

static

Template:

subterm(Subterm,Term)

Mode and number of proofs:

subterm(?term,+term) - zero_or_more

valid/1

Term is valid.

Compilation flags:

static

Template:

valid(Term)

Mode and number of proofs:

valid(@nonvar) - zero_or_one

check/1

Checks if a term is valid. Throws an exception if the term is not valid.

Compilation flags:

static

Template:

check(Term)

Mode and number of proofs:

check(@nonvar) - one

variant/2

Each term is a variant of the other (i.e., they are structurally equivalent).

Compilation flags:

static

Template:

variant(Term1,Term2)

Mode and number of proofs:

variant(@term,@term) - zero_or_one

variables/2

Returns a list of all term variables (ordered as found when doing a depth-first, left-to-right traversal of Term). Deprecated. Use the standard term_variables/2 predicate instead.

Compilation flags:

static

Template:

variables(Term,List)

Mode and number of proofs:

variables(@term,-list) - one

singletons/2

Returns a list of all term singleton variables (ordered as found when doing a depth-first, left-to-right traversal of Term).

Compilation flags:

static

Template:

singletons(Term,Singletons)

Mode and number of proofs:

singletons(@term,-list) - one

numbervars/3

Grounds a term by replacing all variables with '\$VAR'(N) terms with N starting at From. The Next argument is unified with the next value for N after binding all variables.

Compilation flags:

static

Template:

numbervars(Term,From,Next)

Mode and number of proofs:

numbervars(?term,+integer,?integer) - zero_or_one

numbervars/1

Grounds a term by replacing all variables with '\$VAR'(N) terms with N starting at 0.

Compilation flags:

static

Template:

numbervars(Term)

Mode and number of proofs:

numbervars(?term) - zero_or_one

varnumbers/3

Replaces all '\$VAR'(N) sub-terms in a term with fresh variables for all values of N grater or equal to From. Variables in Term are shared with Copy.

Compilation flags:

static

Template:

varnumbers(Term,From,Copy)

Mode and number of proofs:

varnumbers(@term,+integer,?term) - zero_or_one

varnumbers/2

Replaces all '\$VAR'(N) sub-terms in a term with fresh variables for all values of N grater or equal to 0. Variables in Term are shared with Copy.

Compilation flags:

static

Template:

varnumbers(Term,Copy)

Mode and number of proofs:

varnumbers(@term,?term) - zero_or_one

Protected predicates

(no local declarations; see entity ancestors if any)

Private predicates

(no local declarations; see entity ancestors if any)

Operators

(none)

➡ See also

[term](#)

object

1.72.21 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: 2:5:1

Date: 2024-09-26

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: `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.
- 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.

- `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.
- `var_or(Type)` notes: Allows checking if a term is either a variable or a valid value of the given type.
- `ground(Type)` notes: Allows checking if a term is ground and a valid value of the given type.
- `types(Types)` notes: Allows checking if a term is a valid value for one of the types in a list of types.
- `constrain(Type, Closure)` notes: Allows checking if a term is a valid value for the given type and satisfies the given closure.
- `type` notes: Allows checking if a term is a valid 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`, `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.

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

1.72.22 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

1.72.23 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

1.73 ulid

object

1.73.1 ulid

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

Availability:

`logtalk__load(ulid(loader))`

Author: Paulo Moura

Version: 1:0:0

Date: 2023-05-19

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`, `uuid`, `uuid(Representation)`, `ids`, `ids(Representation,Bytes)`

object

1.73.2 `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: 1:0:0

Date: 2023-05-19

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`, `uuid(Representation)`, `uuid`, `ids`, `ids(Representation,Bytes)`

protocol

1.73.3 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

1.73.4 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`

1.74 union_find

object

1.74.1 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`

1.74.2 union_find_protocol

Union-find data structure protocol.

Availability:

`logtalk_load(union_find(loader))`

Author: José Antonio Ríaza 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](#)

1.75 uuid

object

1.75.1 uuid

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

Availability:

`logtalk_load(uuid(loader))`

Author: Paulo Moura

Version: 0:2:0

Date: 2022-11-23

Compilation flags:

`static, context_switching_calls`

Extends:

`public uuid(atom)`

Remarks:

(none)

Inherited public predicates:

`random_node/1 uuid_null/1 uuid_v1/2 uuid_v4/1`

- Public predicates
- Protected predicates
- Private predicates
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

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)`, `ulid`, `ulid(Representation)`, `ids`, `ids(Representation,Bytes)`

object

1.75.2 `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: 0:5:0

Date: 2022-11-23

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_null/1 uuid_v1/2 uuid_v4/1

- Public predicates
- Protected predicates
- Private predicates
- Operators

Public predicates

(no local declarations; see entity ancestors if any)

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, ulid, ulid(Representation), ids, ids(Representation,Bytes)

protocol

1.75.3 uuid_protocol

Universally unique identifier (UUID) generator protocol.

Availability:

`logtalk_load(uuid(loader))`

Author: Paulo Moura

Version: 0:3:0

Date: 2021-03-13

Compilation flags:

`static`

Dependencies:

`(none)`

Remarks:

`(none)`

Inherited public predicates:

`(none)`

- Public predicates
 - `uuid_v1/2`
 - `uuid_v4/1`
 - `uuid_null/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),--ground) - one`

`uuid_v4/1`

Returns a version 4 UUID.

Compilation flags:

`static`

Template:

`uuid_v4(UUID)`

Mode and number of proofs:

`uuid_v4(--ground) - one`

`uuid_null/1`

Returns the null UUID.

Compilation flags:

`static`

Template:

`uuid_null(UUID)`

Mode and number of proofs:

`uuid_null(--ground) - 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)

1.76 verdi_neruda

object

1.76.1 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

1.76.2 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`

1.76.3 `best_first`

Best-first framework for general logic programs.

Availability:

`logtalk_load(verdi_neruda(loader))`

Author: Victor Lagerkvist

Version: 1:1:0
Date: 2019-03-08

Compilation flags:
static

Implements:
public `interpreterp`

Uses:
`counter`
`minheap`

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

1.76.4 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

1.76.5 `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

1.76.6 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

1.76.7 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

1.76.8 `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

1.76.9 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

1.76.10 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

1.76.11 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

1.76.12 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

1.76.13 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

1.76.14 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

1.76.15 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

1.76.16 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

1.76.17 [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

1.76.18 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

1.76.19 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

1.76.20 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)

1.77 wrapper

object

1.77.1 wrapper

Adviser tool for porting and wrapping plain Prolog applications.

Availability:

```
logtalk_load(wrapper(loader))
```

Author: Paulo Moura

Version: 0:12:3

Date: 2025-10-28

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(Files)` 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)

1.78 xml_parser

object

1.78.1 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 pcdat/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)

1.79 zippers

protocol

1.79.1 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

1.79.2 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/arbitrary/`
- 2.9** `library/assignvars/`
- 2.10** `library/base64/`
- 2.11** `library/cbor/`
- 2.12** `library/coroutining/`
- 2.13** `library/csv/`
- 2.14** `library/dates/`
- 2.15** `library/dependents/`
- 2.16** `library/dictionaries/`
- 2.17** `library/dif/`
- 2.18** `library/edcg/`
- 2.19** `library/events/`
- 2.20** `library/expand_library_alias_paths/`
- 2.1.** `contributions/flags/`
- 2.21** `library/expecteds/`

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 `absolute_file_name/2`

- `osp`

4.13 `activate_debug_handler/1`

- `logtalk`

4.14 activate_monitor/0

- monitorp

4.15 active_debug_handler/1

- logtalk

4.16 add/1

- registries

4.17 add/2

- registries

4.18 add/3

- difflist
- registries

4.19 addDependent/1

- subject

4.20 after/2

- intervalp

4.21 after/3

- monitoring

4.22 all/0

- `code_metric`
- `dead_code_scanner`
- `lgtdocp`

4.23 all/1

- `code_metric`
- `dead_code_scanner`
- `lgtdocp`

4.24 all_files/0

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

4.25 all_files/1

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

4.26 all_libraries/0

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

4.27 all_libraries/1

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

4.28 all_score/1

- code_metric

4.29 ancestor/1

- hierarchyp

4.30 ancestors/1

- hierarchyp

4.31 apis/0

- help_info_support

4.32 apis/1

- help_info_support

4.33 append/2

- listp

4.34 append/3

- listp
- queuep
- varlistp

4.35 apply/2

- zipperp

4.36 apply/4

- dictionaryp

4.37 approximately_equal/2

- lgtunit
- number

4.38 approximately_equal/3

- lgtunit
- number

4.39 arbitrary/1

- arbitrary

4.40 arbitrary/2

- arbitrary

4.41 archive/1

- registry_protocol

4.42 arithmetic_mean/2

- statisticsp

4.43 array_list/2

- java_utils_protocol

4.44 array__to__list/2

- java__utils__protocol

4.45 array__to__terms/2

- java__utils__protocol

4.46 array__to__terms/3

- java__utils__protocol

4.47 as__curly__bracketed/2

- dictionaryp
- nested_dictionary_protocol

4.48 as__dictionary/2

- dictionaryp

4.49 as__difflist/2

- list

4.50 as__heap/2

- heapp

4.51 as__list/2

- dictionaryp
- difflist
- heapp
- queuep
- setp

4.52 as__nested__dictionary/2

- nested_dictionary_protocol

4.53 as__set/2

- setp

4.54 ask__question/5

- logtalk

4.55 assertion/1

- assertions(Mode)
- lgtunit

4.56 assertion/2

- assertions(Mode)
- lgtunit

4.57 assignable/1

- assignvarsp

4.58 assignable/2

- assignvarsp

4.59 available/0

- packs

4.60 available/1

- packs

4.61 available/2

- packs

4.62 average/2

- numberlistp

4.63 average_deviation/3

- statisticsp

4.64 before/2

- intervalp

4.65 before/3

- monitoring

4.66 bench_goal/1

- databasep

4.67 benchmark/2

- lgtunit

4.68 benchmark/3

- lgtunit

4.69 benchmark/4

- lgtunit

4.70 benchmark_reified/3

- lgtunit

4.71 bernoulli/2

- sampling_protocol

4.72 beta/3

- sampling_protocol

4.73 between/3

- integer
- random_protocol

4.74 between/4

- float

4.75 binomial/3

- sampling_protocol

4.76 bit//1

- `number__grammars(Format)`

4.77 bits//1

- `number__grammars(Format)`

4.78 blank//0

- `blank__grammars(Format)`

4.79 blanks//0

- `blank__grammars(Format)`

4.80 body__pred/1

- `metagol`

4.81 branch/2

- `git__protocol`

4.82 built__in__directive/4

- `help`

4.83 built__in__flag/2

- `flags`

4.84 `built_in_method/4`

- `help`

4.85 `built_in_non_terminal/4`

- `help`

4.86 `built_in_predicate/4`

- `help`

4.87 `calendar_month/3`

- `iso8601`

4.88 `call_with_timeout/2`

- `timeout`

4.89 `call_with_timeout/3`

- `timeout`

4.90 `cat/2`

- `maybe`

4.91 `change_directory/1`

- `osp`

4.92 changed/0

- subject

4.93 changed/1

- subject

4.94 chebyshev_distance/3

- numberlistp

4.95 chebyshev_norm/2

- numberlistp

4.96 check/1

- term p
- varlistp

4.97 check/2

- type

4.98 check/3

- type

4.99 check_option/1

- options_protocol

4.100 check_options/1

- options_protocol

4.101 chi_squared/2

- sampling_protocol

4.102 chr_is/2

- toychrdb

4.103 chr_no_spy/1

- toychrdb

4.104 chr_nospy/0

- toychrdb

4.105 chr_notrace/0

- toychrdb

4.106 chr_option/2

- toychrdb

4.107 chr_spy/1

- toychrdb

4.108 chr_trace/0

- toychrdb

4.109 circular_uniform_cartesian/3

- sampling_protocol

4.110 circular_uniform_polar/3

- sampling_protocol

4.111 class/1

- class_hierarchy

4.112 classes/1

- class_hierarchy

4.113 clause/5

- ports_profiler

4.114 clause_location/6

- ports_profiler

4.115 clean/0

- packs
- registries

4.116 clean/1

- packs
- registries

4.117 clean/2

- packs

4.118 clone/1

- cloning
- registry_protocol

4.119 clone/3

- dictionaryp

4.120 clone/4

- dictionaryp

4.121 coefficient_of_variation/2

- statisticsp

4.122 command_line_arguments/1

- osp

4.123 commit_author/2

- git_protocol

4.124 `commit_date/2`

- `git_protocol`

4.125 `commit_hash/2`

- `git_protocol`

4.126 `commit_hash_abbreviated/2`

- `git_protocol`

4.127 `commit_log/3`

- `git_protocol`

4.128 `commit_message/2`

- `git_protocol`

4.129 `compile_aux_clauses/1`

- `logtalk`

4.130 `compile_predicate_heads/4`

- `logtalk`

4.131 `compile_predicate_indicators/3`

- `logtalk`

4.132 completion/2

- help

4.133 completions/2

- help

4.134 connect/1

- redis

4.135 connect/3

- redis

4.136 console/1

- redis

4.137 contains/2

- intervalp

4.138 control//0

- blank_grammars(Format)

4.139 control_construct/4

- help

4.140 controls//0

- blank_grammars(Format)

4.141 copy_file/2

- osp

4.142 counter/2

- counters
- mutations_store

4.143 cover/1

- lgtunit

4.144 cpu_time/1

- osp
- timep

4.145 current/2

- zipperp

4.146 data/0

- ports_profiler

4.147 data/1

- ports_profiler

4.148 data/2

- ports__profiler

4.149 date/4

- iso8601

4.150 date/5

- iso8601

4.151 date/6

- iso8601

4.152 date/7

- iso8601

4.153 date_string/3

- iso8601

4.154 date_time/7

- osp

4.155 days_in_month/3

- datep

4.156 deactivate_debug_handler/0

- logtalk

4.157 debug/0

- debuggerp

4.158 debug_handler/1

- logtalk

4.159 debug_handler/3

- logtalk

4.160 debugging/0

- debuggerp

4.161 debugging/1

- debuggerp

4.162 decide/1

- fcube

4.163 decide/2

- fcube

4.164 decode__exception/2

- java__utils__protocol

4.165 decode__exception/3

- java__utils__protocol

4.166 decompile__predicate__heads/4

- logtalk

4.167 decompile__predicate__indicators/4

- logtalk

4.168 decompose__file__name/3

- osp

4.169 decompose__file__name/4

- osp

4.170 decrement__counter/1

- counters

4.171 default__option/1

- options__protocol
- wrapper

4.172 default_options/1

- options_protocol
- wrapper

4.173 define_log_file/2

- loggingp

4.174 defined/4

- registries

4.175 defined_flag/6

- flags

4.176 del_monitors/0

- event_registryp

4.177 del_monitors/4

- event_registryp

4.178 del_spy_points/4

- monitorp

4.179 delete/0

- registries

4.180 delete/1

- registries

4.181 delete/2

- registries

4.182 delete/3

- listp
- setp
- varlistp

4.183 delete/4

- dictionaryp
- heapp

4.184 delete_all_after/2

- zipperp

4.185 delete_all_after_and_unzip/2

- zipperp

4.186 delete_all_before/2

- zipperp

4.187 delete_all_before_and_unzip/2

- zipperp

4.188 delete__and__next/2

- zipperp

4.189 delete__and__previous/2

- zipperp

4.190 delete__and__unzip/2

- zipperp

4.191 delete__directory/1

- osp

4.192 delete__directory__and__contents/1

- osp

4.193 delete__directory__contents/1

- osp

4.194 delete__file/1

- osp

4.195 delete__in/4

- nested_dictionary_protocol

4.196 delete_matches/3

- listp

4.197 delete_max/4

- dictionaryp

4.198 delete_min/4

- dictionaryp

4.199 depends/1

- packs
- subject

4.200 depends/2

- packs

4.201 depends/3

- packs

4.202 depth/2

- term p

4.203 descendant/1

- hierarchyp

4.204 descendant_class/1

- class_hierarchy

4.205 descendant_classes/1

- class_hierarchy

4.206 descendant_instance/1

- class_hierarchy

4.207 descendant_instances/1

- class_hierarchy

4.208 descendants/1

- hierarchy

4.209 describe/1

- packs
- registries

4.210 describe/2

- packs

4.211 description/1

- pack_protocol
- registry_protocol

4.212 deterministic/1

- lgtunit

4.213 deterministic/2

- lgtunit

4.214 diagram_description/1

- diagram(Format)

4.215 diagram_name_suffix/1

- diagram(Format)

4.216 dif/1

- coroutining
- dif

4.217 dif/2

- coroutining
- dif

4.218 digit//1

- number_grammars(Format)

4.219 digits//1

- number_grammars(Format)

4.220 directories/1

- lgtdocp
- wrapper

4.221 directories/2

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

4.222 directories/3

- diagram(Format)
- diagrams(Format)

4.223 directory/1

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

4.224 directory/2

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

4.225 directory/3

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

4.226 directory_exists/1

- `directory_exists`

4.227 directory_files/2

- `directory_files`

4.228 directory_files/3

- `directory_files`

4.229 directory_score/2

- `code_metric`

4.230 dirichlet/2

- `sampling_protocol`

4.231 disable/1

- `debug_messages`

4.232 disable/2

- `debug_messages`

4.233 `disable_logging/1`

- `loggingp`

4.234 `disconnect/1`

- `redis`

4.235 `disjoint/2`

- `setp`

4.236 `disjoint_sets/2`

- `union_find_protocol`

4.237 `doc_goal/1`

- `doclet`

4.238 `dot//1`

- `number_grammars(Format)`

4.239 `dowhile/2`

- `loopp`

4.240 `drop/3`

- `listp`

4.241 during/2

- intervalp

4.242 easter_day/3

- iso8601

4.243 edge/6

- graph_language_protocol

4.244 edge_case/2

- arbitrary

4.245 either/3

- expected(Expected)

4.246 empty/1

- dictionaryp
- heapp
- listp
- nested_dictionary_protocol
- optional
- queuep
- setp
- varlistp

4.247 enable/1

- debug_messages

4.248 enable/2

- debug_messages

4.249 enable_logging/1

- loggingp

4.250 enabled/1

- debug_messages

4.251 enabled/2

- debug_messages

4.252 ensure_directory/1

- osp

4.253 ensure_file/1

- osp

4.254 entity/1

- code_metric
- dead_code_scanner
- help
- xref_diagram(Format)

4.255 entity/2

- xref_diagram(Format)

4.256 entity_info_pair_score_hook/3

- doc_metric

4.257 entity_info_score_hook/2

- doc_metric

4.258 entity_predicates_weights_hook/2

- doc_metric

4.259 entity_prefix/2

- logtalk

4.260 entity_score/2

- code_metric

4.261 enumerate/2

- random_protocol

4.262 environment_variable/2

- osp

4.263 epsilon/1

- lgtunit

4.264 equal/2

- intervalp
- setp

4.265 erase/1

- recorded__database__core

4.266 essentially_equal/3

- lgtunit
- number

4.267 euclidean_distance/3

- numberlistp

4.268 euclidean_norm/2

- numberlistp

4.269 exclude/3

- metap

4.270 execution_context/7

- logtalk

4.271 `expand_library_path/2`

- `logtalk`

4.272 `expected/1`

- `expected(Expected)`

4.273 `expecteds/2`

- `either`

4.274 `explain//1`

- `tutor`

4.275 `exponential/2`

- `sampling_protocol`

4.276 `extension/1`

- `proto_hierarchy`

4.277 `extensions/1`

- `proto_hierarchy`

4.278 `failed_test_reason//1`

- `lgtunit_messages`

4.279 false/1

- java_utils_protocol

4.280 fcube/0

- fcube

4.281 file/1

- code_metric
- dead_code_scanner
- entity_diagram(Format)
- lgtdocp
- wrapper

4.282 file/2

- code_metric
- dead_code_scanner
- entity_diagram(Format)
- lgtdocp
- wrapper

4.283 file_exists/1

- osp

4.284 file_footer/3

- graph_language_protocol

4.285 file_header/3

- graph_language_protocol

4.286 file_modification_time/2

- osp

4.287 file_permission/2

- osp

4.288 file_score/2

- code_metric

4.289 file_size/2

- osp

4.290 file_to_bytes/2

- reader

4.291 file_to_bytes/3

- reader

4.292 file_to_chars/2

- reader

4.293 file_to_chars/3

- reader

4.294 file_to_codes/2

- reader

4.295 file_to_codes/3

- reader

4.296 file_to_terms/2

- reader

4.297 file_to_terms/3

- reader

4.298 file_type_extension/2

- logtalk

4.299 files/1

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

4.300 files/2

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

4.301 files/3

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

4.302 filter/2

- `optional(Optional)`

4.303 find/4

- `union_find_protocol`

4.304 find/5

- `union_find_protocol`

4.305 findall_member/4

- `metap`

4.306 findall_member/5

- `metap`

4.307 finished_by/2

- intervalp

4.308 finishes/2

- intervalp

4.309 fisher/3

- sampling_protocol

4.310 flag_group_chk/1

- flags

4.311 flag_groups/1

- flags

4.312 flat_map/2

- expected(Expected)
- optional(Optional)

4.313 flatten/2

- listp
- varlistp

4.314 float//1

- number_grammars(Format)

4.315 fold_left/4

- metap

4.316 fold_left_1/3

- metap

4.317 fold_right/4

- metap

4.318 fold_right_1/3

- metap

4.319 fordownto/3

- looppp

4.320 fordownto/4

- looppp

4.321 fordownto/5

- looppp

4.322 foreach/3

- looppp

4.323 foreach/4

- loopp

4.324 format/2

- format

4.325 format/3

- format

4.326 format_entity_score//2

- code_metric

4.327 format_object/1

- diagram(Format)

4.328 format_to_atom/3

- term_io_protocol

4.329 format_to_chars/3

- term_io_protocol

4.330 format_to_chars/4

- term_io_protocol

4.331 `format_to_codes/3`

- `term_io_protocol`

4.332 `format_to_codes/4`

- `term_io_protocol`

4.333 `forto/3`

- `loopp`

4.334 `forto/4`

- `loopp`

4.335 `forto/5`

- `loopp`

4.336 `forward/1`

- `forwarding`

4.337 `forward/2`

- `zipperp`

4.338 `forward/3`

- `zipperp`

4.339 fractile/3

- statisticsp

4.340 freeze/2

- coroutining

4.341 from_generator/2

- expected
- optional

4.342 from_generator/3

- expected
- optional

4.343 from_generator/4

- expected

4.344 from_goal/2

- expected
- optional

4.345 from_goal/3

- expected
- optional

4.346 from_goal/4

- expected

4.347 frozen/2

- coroutining

4.348 full_device_path/1

- osp

4.349 func_test/3

- metagol

4.350 functional/0

- metagol

4.351 gamma/3

- sampling_protocol

4.352 generate/1

- ids(Representation,Bytes)
- ulid_protocol

4.353 generate/2

- base64
- base64url
- cbor(StringRepresentation)
- html
- json_lines_protocol
- json_protocol
- ulid_protocol

4.354 generate/8

- ulid_protocol

4.355 genint/2

- genint_core

4.356 gensym/2

- gensym_core

4.357 geometric/2

- sampling_protocol

4.358 geometric_mean/2

- statisticsp

4.359 get/1

- optional(Optional)

4.360 get_field/2

- java_access_protocol

4.361 get_flag_value/2

- flags

4.362 get_seed/1

- arbitrary
- pseudo_random_protocol

4.363 gnu/0

- fcube

4.364 goal_expansion/2

- expanding

4.365 graph_footer/5

- graph_language_protocol

4.366 graph_header/5

- graph_language_protocol

4.367 ground/1

- term

4.368 group_by_key/2

- pairs

4.369 group_consecutive_by_key/2

- pairs

4.370 group_sorted_by_key/2

- pairs

4.371 guess_arity/2

- csv_protocol

4.372 guess_separator/2

- csv_protocol

4.373 gumbel/3

- sampling_protocol

4.374 hamming_distance/3

- listp

4.375 handbook/0

- help_info_support

4.376 handbook/1

- help_info_support

4.377 harmonic_mean/2

- statisticsp

4.378 head/2

- `queuep`

4.379 head__pred/1

- `metagol`

4.380 help/0

- `help`
- `packs__common`

4.381 hex__digit//1

- `number__grammars(Format)`

4.382 hex__digits//1

- `number__grammars(Format)`

4.383 home/1

- `pack__protocol`
- `registry__protocol`

4.384 hypergeometric/4

- `sampling__protocol`

4.385 ibk/3

- `metagol`

4.386 if_empty/1

- optional(Optional)

4.387 if_expected/1

- expected(Expected)

4.388 if_expected_or_else/2

- expected(Expected)

4.389 if_present/1

- optional(Optional)

4.390 if_present_or_else/2

- optional(Optional)

4.391 if_unexpected/1

- expected(Expected)

4.392 include/3

- metap

4.393 increase/1

- counter

4.394 increment/0

- counter

4.395 increment_counter/1

- counters

4.396 init/0

- shell(Interpreters)

4.397 init_log_file/2

- loggingp

4.398 inorder/2

- bintree

4.399 insert/3

- setp

4.400 insert/4

- dictionaryp
- heapp

4.401 insert_after/3

- zipperp

4.402 insert_all/3

- heapp
- setp

4.403 insert_before/3

- zipperp

4.404 insert_in/4

- nested_dictionary_protocol

4.405 install/1

- packs

4.406 install/2

- packs

4.407 install/3

- packs

4.408 install/4

- packs

4.409 installed/0

- packs

4.410 installed/1

- packs

4.411 installed/3

- packs

4.412 installed/4

- packs

4.413 instance/1

- class_hierarchy

4.414 instance/2

- recorded_database_core

4.415 instances/1

- class_hierarchy

4.416 integer//1

- number_grammars(Format)

4.417 internal_os_path/2

- osp

4.418 intersect/2

- setp

4.419 intersection/2

- dictionaryp

4.420 intersection/3

- dictionaryp
- setp

4.421 intersection/4

- setp

4.422 invoke/1

- java__access__protocol

4.423 invoke/2

- java__access__protocol

4.424 ipv4//1

- ip_grammars(Format)

4.425 ipv6//1

- ip_grammars(Format)

4.426 `is__absolute__file__name/1`

- `osp`

4.427 `is__alpha/1`

- `characterp`

4.428 `is__alphanumeric/1`

- `characterp`

4.429 `is__ascii/1`

- `characterp`

4.430 `is__bin__digit/1`

- `characterp`

4.431 `is__control/1`

- `characterp`

4.432 `is__dec__digit/1`

- `characterp`

4.433 `is__empty/0`

- `optional(Optional)`

4.434 `is__end__of__line/1`

- `characterp`

4.435 `is__expected/0`

- `expected(Expected)`

4.436 `is__false/1`

- `java__utils__protocol`

4.437 `is__hex__digit/1`

- `characterp`

4.438 `is__layout/1`

- `characterp`

4.439 `is__letter/1`

- `characterp`

4.440 `is__lower__case/1`

- `characterp`

4.441 `is__newline/1`

- `characterp`

4.442 is__null/1

- java_utils_protocol

4.443 is__object/1

- java_utils_protocol

4.444 is__octal__digit/1

- characterp

4.445 is__period/1

- characterp

4.446 is__present/0

- optional(Optional)

4.447 is__punctuation/1

- characterp

4.448 is__quote/1

- characterp

4.449 is__true/1

- java_utils_protocol

4.450 is__unexpected/0

- expected(Expected)

4.451 is__upper__case/1

- characterp

4.452 is__void/1

- java_utils_protocol

4.453 is__vowel/1

- characterp

4.454 is__white__space/1

- characterp

4.455 iterator__element/2

- java_utils_protocol

4.456 join/3

- queuep

4.457 join__all/3

- queuep

4.458 jump/3

- queuep

4.459 jump_all/3

- queuep

4.460 jump_all_block/3

- queuep

4.461 key/2

- pairs

4.462 keys/2

- dictionaryp
- pairs

4.463 keys_values/3

- pairs

4.464 keysort/2

- listp

4.465 kurtosis/2

- statisticsp

4.466 language_object/2

- graph_language_registry

4.467 last/2

- listp
- varlistp

4.468 leaf/1

- hierarchyp

4.469 leaf_class/1

- class_hierarchyp

4.470 leaf_classes/1

- class_hierarchyp

4.471 leaf_instance/1

- class_hierarchyp

4.472 leaf_instances/1

- class_hierarchyp

4.473 leap_year/1

- datep
- iso8601

4.474 learn/0

- metagol_example_protocol

4.475 learn/1

- metagol_example_protocol

4.476 learn/2

- metagol

4.477 learn/3

- metagol

4.478 learn_seq/2

- metagol

4.479 learn_with_timeout/4

- metagol

4.480 leash/1

- debuggerp

4.481 leashing/1

- debuggerp

4.482 least_common_multiple/2

- numberlistp

4.483 leaves/1

- hierarchyp

4.484 length/2

- listp
- queuep
- varlistp

4.485 libraries/1

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

4.486 libraries/2

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

4.487 libraries/3

- diagram(Format)
- diagrams(Format)

4.488 library/0

- help

4.489 library/1

- code__metric
- dead_code_scanner
- diagram(Format)
- diagrams(Format)
- help
- lgtdocp

4.490 library/2

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

4.491 library__score/2

- code__metric

4.492 license/1

- pack_protocol

4.493 line_to_chars/2

- reader

4.494 line_to_chars/3

- reader

4.495 line_to_codes/2

- reader

4.496 line_to_codes/3

- reader

4.497 lint/0

- packs
- registries

4.498 lint/1

- packs
- registries

4.499 lint/2

- packs

4.500 list/0

- registries

4.501 list_to_array/2

- java_utils_protocol

4.502 listing/0

- listing

4.503 listing/1

- listing

4.504 loaded_file/1

- logtalk

4.505 loaded_file_property/2

- logtalk
- modules_diagram_support

4.506 loaded_files_topological_sort/1

- logtalk

4.507 loaded_files_topological_sort/2

- logtalk

4.508 log/3

- debuggerp

4.509 log_event/2

- loggingp

4.510 log_file/2

- loggingp

4.511 logging/1

- loggingp

4.512 logging/3

- debuggerp

4.513 logistic/3

- sampling_protocol

4.514 lognormal/3

- sampling_protocol

4.515 logseries/2

- sampling_protocol

4.516 logtalk_packs/0

- packs_common

4.517 logtalk_packs/1

- packs_common

4.518 lookup/2

- dictionaryp

4.519 lookup/3

- dictionaryp

4.520 lookup_in/3

- nested_dictionary_protocol

4.521 lower_upper/2

- characterp

4.522 magic/2

- magic

4.523 magicise/4

- magic

4.524 make_directory/1

- osp

4.525 make_directory_path/1

- osp

4.526 `make__set/3`

- `union_find_protocol`

4.527 `man/1`

- `help_info_support`

4.528 `manhattan_distance/3`

- `numberlistp`

4.529 `manhattan_norm/2`

- `numberlistp`

4.530 `manuals/0`

- `help`

4.531 `map/2`

- `dictionaryp`
- `expected(Expected)`
- `metap`
- `optional(Optional)`
- `queuep`

4.532 `map/3`

- `dictionaryp`
- `metap`
- `pairs`
- `queuep`

4.533 map/4

- `metap`

4.534 map/5

- `metap`

4.535 map/6

- `metap`

4.536 map/7

- `metap`

4.537 map/8

- `metap`

4.538 map_element/2

- `java_utils_protocol`

4.539 map_reduce/5

- `metap`

4.540 max/2

- `listp`
- `numberlistp`
- `statisticsp`

4.541 max/3

- dictionaryp

4.542 max_clauses/1

- metagol

4.543 max_inv_preds/1

- metagol

4.544 max_size/1

- arbitrary

4.545 maybe/0

- random_protocol

4.546 maybe/1

- random_protocol

4.547 maybe/2

- random_protocol

4.548 maybe_call/1

- random_protocol

4.549 maybe_call/2

- random_protocol

4.550 mean_deviation/2

- statisticsp

4.551 median/2

- numberlistp
- statisticsp

4.552 median_deviation/2

- statisticsp

4.553 meets/2

- intervalp

4.554 member/2

- listp
- random_protocol
- setp

4.555 memberchk/2

- listp
- setp
- varlistp

4.556 merge/3

- heapp

4.557 message_hook/4

- logtalk

4.558 message_prefix_file/6

- logtalk

4.559 message_prefix_stream/4

- logtalk

4.560 message_tokens//2

- logtalk

4.561 met_by/2

- intervalp

4.562 meta_type/3

- type

4.563 metarule/6

- metagol

4.564 `metarule_next_id/1`

- `metagol`

4.565 `min/2`

- `listp`
- `numberlistp`
- `statisticsp`

4.566 `min/3`

- `dictionaryp`

4.567 `min_clauses/1`

- `metagol`

4.568 `min_max/3`

- `numberlistp`
- `statisticsp`

4.569 `modes/2`

- `numberlistp`
- `statisticsp`

4.570 `module_property/2`

- `modules_diagram_support`

4.571 monitor/1

- event_registryp

4.572 monitor/4

- event_registryp

4.573 monitor_activated/0

- monitorp

4.574 monitored/1

- event_registryp

4.575 monitors/1

- event_registryp

4.576 msort/2

- listp

4.577 msort/3

- listp

4.578 mutation/3

- mutations
- mutations_store

4.579 name/1

- pack_protocol
- registry_protocol

4.580 name_of_day/3

- datep

4.581 name_of_month/3

- datep

4.582 natural//1

- number_grammars(Format)

4.583 new/1

- java_access_protocol
- nested_dictionary_protocol
- streamvars
- termp

4.584 new/2

- java_access_protocol
- streamvars
- union_find_protocol

4.585 new/3

- intervalp

4.586 new_line//0

- blank_grammars(Format)

4.587 new_lines//0

- blank_grammars(Format)

4.588 next/2

- zipperp

4.589 next/3

- zipperp

4.590 next/4

- dictionaryp

4.591 nextto/3

- listp
- varlistp

4.592 node/7

- graph_language_protocol

4.593 nodebug/0

- debuggerp

4.594 nolog/3

- debuggerp

4.595 nologall/0

- debuggerp

4.596 non_blank//1

- blank_grammars(Format)

4.597 non_blanks//1

- blank_grammars(Format)

4.598 normal/3

- sampling_protocol

4.599 normal_element/2

- html

4.600 normalize_range/2

- numberlistp

4.601 normalize_range/4

- numberlistp

4.602 normalize_scalar/2

- numberlistp

4.603 normalize_unit/2

- numberlistp

4.604 nospy/1

- debuggerp

4.605 nospy/3

- debuggerp

4.606 nospy/4

- debuggerp

4.607 nospyall/0

- debuggerp

4.608 note/2

- registry_protocol

4.609 note/3

- pack_protocol

4.610 notrace/0

- debuggerp

4.611 now/3

- timep

4.612 nth0/3

- listp
- varlistp

4.613 nth0/4

- listp
- varlistp

4.614 nth1/3

- listp
- varlistp

4.615 nth1/4

- listp
- varlistp

4.616 null/1

- java_utils_protocol

4.617 null_device_path/1

- osp

4.618 number//1

- number_grammars(Format)

4.619 number_of_tests/1

- lgtunit

4.620 numbervars/1

- term_p

4.621 numbervars/3

- term_p

4.622 occurrences/2

- list_p

4.623 occurrences/3

- list_p

4.624 occurs/2

- term_p

4.625 of/2

- optional

4.626 of_expected/2

- expected

4.627 of_unexpected/2

- expected

4.628 one_or_more//0

- sequence_grammars

4.629 one_or_more//1

- sequence_grammars

4.630 one_or_more//2

- sequence_grammars

4.631 operating_system_machine/1

- osp

4.632 operating_system_name/1

- osp

4.633 `operating_system_release/1`

- `osp`

4.634 `operating_system_type/1`

- `osp`

4.635 `option/2`

- `options_protocol`

4.636 `option/3`

- `options_protocol`

4.637 `or/2`

- `optional(Optional)`

4.638 `or_else/2`

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

4.639 `or_else_call/2`

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

4.640 `or_else_fail/1`

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

4.641 or_else_get/2

- expected(Expected)
- optional(Optional)

4.642 or_else_throw/1

- expected(Expected)

4.643 or_else_throw/2

- optional(Optional)

4.644 orphaned/0

- packs

4.645 orphaned/2

- packs

4.646 outdated/0

- packs

4.647 outdated/1

- packs

4.648 outdated/2

- packs

4.649 outdated/4

- packs

4.650 outdated/5

- packs

4.651 output_file_name/2

- graph_language_protocol

4.652 overlapped_by/2

- intervalp

4.653 overlaps/2

- intervalp

4.654 parent/1

- proto_hierarchy

4.655 parenthesis/2

- characterp

4.656 parents/1

- proto_hierarchy

4.657 parse/2

- base64
- base64url
- cbor(StringRepresentation)
- json_lines_protocol
- json_protocol
- xml

4.658 parse/3

- xml

4.659 parse_domain/2

- pddl

4.660 parse_domain/3

- pddl

4.661 parse_problem/2

- pddl

4.662 parse_problem/3

- pddl

4.663 partial_map/4

- rbtrees

4.664 partition/3

- either

4.665 partition/4

- metap

4.666 partition/5

- listp

4.667 partition/6

- metap

4.668 path_concat/3

- osp

4.669 permutation/2

- listp
- random_protocol
- varlistp

4.670 pid/1

- osp

4.671 pin/0

- packs_common

4.672 pin/1

- packs_common

4.673 pinned/1

- packs_common

4.674 plus/3

- integer

4.675 poisson/2

- sampling_protocol

4.676 port/5

- ports_profiler

4.677 portray_clause/1

- listing

4.678 postorder/2

- bintree

4.679 power/2

- sampling_protocol

4.680 power_sequence/4

- integer

4.681 powerset/2

- setp

4.682 pp/1

- xml

4.683 pprint/1

- metagol

4.684 predicate/2

- dead_code_scanner

4.685 predicate_info_pair_score_hook/4

- doc_metric

4.686 predicate_info_score_hook/3

- doc_metric

4.687 predicate_mode_score_hook/3

- doc_metric

4.688 predicate_mode_score_hook/5

- doc_metric

4.689 predicates/2

- dead_code_scanner

4.690 prefix/0

- packs_common

4.691 prefix/1

- packs_common

4.692 prefix/2

- listp
- varlistp

4.693 prefix/3

- listp

4.694 preorder/2

- bintree

4.695 previous/2

- zipperp

4.696 previous/3

- zipperp

4.697 previous/4

- dictionaryp

4.698 print__flags/0

- flags
- flags_validator

4.699 print__flags/1

- flags

4.700 print__message/3

- logtalk

4.701 print__message_token/4

- logtalk

4.702 print__message_tokens/3

- logtalk

4.703 product/2

- numberlistp
- statisticsp

4.704 product/3

- setp

4.705 program_to_clauses/2

- metagol

4.706 proper_prefix/2

- listp

4.707 proper_prefix/3

- listp

4.708 proper_suffix/2

- listp

4.709 proper_suffix/3

- listp

4.710 prove/2

- interpreterp

4.711 prove/3

- interpreterp

4.712 provides/2

- registries

4.713 question_hook/6

- logtalk

4.714 question_prompt_stream/4

- logtalk

4.715 quick_check/1

- lgtunit

4.716 quick_check/2

- lgtunit

4.717 quick_check/3

- lgtunit

4.718 random/1

- random_protocol

4.719 random/3

- random_protocol

4.720 random_node/1

- uuid_protocol

4.721 random_tree/1

- benchmark_generators

4.722 randomize/1

- fast_random
- random

4.723 randseq/4

- random_protocol

4.724 randset/4

- random_protocol

4.725 range/2

- statisticsp

4.726 rdirectories/1

- lgtdocp

4.727 rdirectories/2

- lgtdocp

4.728 rdirectory/1

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

4.729 rdirectory/2

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

4.730 rdirectory/3

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

4.731 rdirectory__score/2

- `code__metric`

4.732 read__file/2

- `csv__protocol`
- `read__file`
- `tsv__protocol`

4.733 read_file/3

- csv_protocol
- tsv_protocol

4.734 read_file_by_line/2

- csv_protocol
- tsv_protocol

4.735 read_file_by_line/3

- csv_protocol
- tsv_protocol

4.736 read_from_atom/2

- term_io_protocol

4.737 read_from_chars/2

- term_io_protocol

4.738 read_from_codes/2

- term_io_protocol

4.739 read_only_device_path/1

- osp

4.740 read_stream/2

- csv_protocol
- tsv_protocol

4.741 read_stream/3

- csv_protocol
- tsv_protocol

4.742 read_stream_by_line/2

- csv_protocol
- tsv_protocol

4.743 read_stream_by_line/3

- csv_protocol
- tsv_protocol

4.744 read_term_from_atom/3

- term_io_protocol

4.745 read_term_from_chars/3

- term_io_protocol

4.746 read_term_from_chars/4

- term_io_protocol

4.747 read_term_from_codes/3

- term_io_protocol

4.748 read_term_from_codes/4

- term_io_protocol

4.749 readme/1

- packs_common

4.750 readme/2

- packs_common

4.751 recorda/2

- recorded_database_core

4.752 recorda/3

- recorded_database_core

4.753 recorded/2

- recorded_database_core

4.754 recorded/3

- recorded_database_core

4.755 recordz/2

- recorded_database_core

4.756 recordz/3

- recorded_database_core

4.757 relative_standard_deviation/2

- statisticsp

4.758 removeDependent/1

- subject

4.759 remove_duplicates/2

- listp
- varlistp

4.760 rename_file/2

- osp

4.761 replace/3

- zipperp

4.762 replace_sub_atom/4

- atom

4.763 rescale/3

- numberlistp

4.764 reset/0

- counter
- debuggerp
- packs_common
- ports_profiler

4.765 reset/1

- ports_profiler

4.766 reset_counter/1

- counters

4.767 reset_counters/0

- counters

4.768 reset_flags/0

- flags

4.769 reset_flags/1

- flags

4.770 reset_genint/0

- genint_core

4.771 reset_genint/1

- genint_core

4.772 reset_gensym/0

- gensym_core

4.773 reset_gensym/1

- gensym_core

4.774 reset_monitor/0

- monitorp

4.775 reset_seed/0

- fast_random
- random

4.776 restore/1

- packs

4.777 restore/2

- packs

4.778 reverse/2

- listp
- varlistp

4.779 rewind/2

- zipperp

4.780 rewind/3

- zipperp

4.781 rlibraries/1

- lgtdocp

4.782 rlibraries/2

- lgtdocp

4.783 rlibrary/1

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

4.784 rlibrary/2

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

4.785 rlibrary_score/2

- code_metric

4.786 rule/2

- databasep

4.787 rule/3

- databasep

4.788 rule/4

- databasep

4.789 run/0

- lgtunit

4.790 run/1

- lgtunit

4.791 run/2

- lgtunit

4.792 run_test_sets/1

- lgtunit

4.793 same_length/2

- listp
- varlistp

4.794 same_length/3

- listp

4.795 save/0

- wrapper

4.796 save/1

- packs
- wrapper

4.797 save/2

- packs

4.798 scalar_product/3

- numberlistp

4.799 scan_left/4

- metap

4.800 scan_left_1/3

- metap

4.801 scan_right/4

- metap

4.802 scan_right_1/3

- metap

4.803 search/1

- packs

4.804 select/3

- listp
- random_protocol
- setp
- varlistp

4.805 select/4

- listp
- random_protocol

4.806 selectchk/3

- listp
- setp

4.807 selectchk/4

- listp

4.808 send/3

- redis

4.809 sequence/3

- integer

4.810 sequence/4

- float
- integer
- random_protocol

4.811 sequence/5

- float

4.812 sequential_occurrences/2

- listp

4.813 sequential_occurrences/3

- listp

4.814 serve/3

- queuep

4.815 set/1

- counter

4.816 set/4

- random_protocol

4.817 set_element/2

- java_utils_protocol

4.818 set_field/2

- java_access_protocol

4.819 set_flag_value/2

- flags

4.820 set_flag_value/3

- flags

4.821 set_monitor/4

- event_registry

4.822 set_seed/1

- arbitrary
- pseudo_random_protocol

4.823 set_spy_point/4

- monitorp

4.824 set_write_max_depth/1

- debuggerp

4.825 setup/0

- packs_common

4.826 shell/1

- osp

4.827 shell/2

- osp

4.828 shell_command/1

- doclet

4.829 shrink/3

- arbitrary

4.830 shrink_sequence/3

- arbitrary

4.831 shrinker/1

- arbitrary

4.832 sign//1

- number__grammars(Format)

4.833 singletons/2

- term_p

4.834 size/2

- dictionary_p
- heap_p
- set_p

4.835 skewness/2

- statistics_p

4.836 sleep/1

- osp

4.837 softmax/2

- numberlist_p

4.838 softmax/3

- numberlist_p

4.839 sort/2

- list_p

4.840 sort/3

- listp

4.841 sort/4

- listp

4.842 source_file_extension/1

- modules_diagram_support

4.843 space//0

- blank_grammars(Format)

4.844 spaces//0

- blank_grammars(Format)

4.845 split/3

- atom

4.846 split/4

- listp

4.847 spy/1

- debuggerp

4.848 spy/3

- debuggerp

4.849 spy/4

- debuggerp

4.850 spy__point/4

- monitorp

4.851 spying/1

- debuggerp

4.852 spying/3

- debuggerp

4.853 spying/4

- debuggerp

4.854 standard__cauchy/3

- sampling__protocol

4.855 standard_deviation/2

- statisticsp

4.856 standard_exponential/1

- sampling_protocol

4.857 standard_gamma/2

- sampling_protocol

4.858 standard_normal/1

- sampling_protocol

4.859 standard_t/2

- sampling_protocol

4.860 start/0

- ports_profiler
- shell

4.861 start_redirect_to_file/2

- dump_trace

4.862 started_by/2

- intervalp

4.863 starts/2

- intervalp

4.864 stop/0

- ports_profiler

4.865 stop_redirect_to_file/0

- dump_trace

4.866 stream_to_bytes/2

- reader

4.867 stream_to_bytes/3

- reader

4.868 stream_to_chars/2

- reader

4.869 stream_to_chars/3

- reader

4.870 stream_to_codes/2

- reader

4.871 stream_to_codes/3

- reader

4.872 stream_to_terms/2

- reader

4.873 stream_to_terms/3

- reader

4.874 subclass/1

- class_hierarchyp

4.875 subclasses/1

- class_hierarchyp

4.876 sublist/2

- listp
- varlistp

4.877 subsequence/3

- listp

4.878 subsequence/4

- listp

4.879 subset/2

- setp

4.880 substitute/4

- listp

4.881 subsumes/2

- term p

4.882 subterm/2

- term p
- xml

4.883 subtract/3

- listp
- setp
- varlistp

4.884 succ/2

- integer

4.885 suffix/2

- listp
- varlistp

4.886 suffix/3

- listp

4.887 `sum/2`

- `numberlistp`
- `statisticsp`

4.888 `superclass/1`

- `class_hierarchyp`

4.889 `superclasses/1`

- `class_hierarchyp`

4.890 `suspend_monitor/0`

- `monitorp`

4.891 `swap/2`

- `random_protocol`

4.892 `swap_consecutive/2`

- `random_protocol`

4.893 `syndiff/3`

- `setp`

4.894 `tab//0`

- `blank_grammars(Format)`

4.895 tabs//0

- blank_grammars(Format)

4.896 take/3

- listp

4.897 temporary_directory/1

- osp

4.898 term_expansion/2

- expanding

4.899 terms_to_array/2

- java_utils_protocol

4.900 test/1

- lgtunit

4.901 time_stamp/1

- osp

4.902 timeout/1

- metagol

4.903 timestamp/2

- ulid_protocol

4.904 timestamp/8

- ulid_protocol

4.905 today/3

- datep

4.906 tolerance_equal/4

- lgtunit
- number

4.907 top/3

- heapp

4.908 top_next/5

- heapp

4.909 trace/0

- debuggerp

4.910 trace_event/2

- logtalk

4.911 transpose/2

- pairs

4.912 triangular/4

- sampling_protocol

4.913 true/1

- java_utils_protocol

4.914 type/1

- type

4.915 unexpected/1

- expected(Expected)

4.916 unexpecteds/2

- either

4.917 uniform/1

- sampling_protocol

4.918 uniform/3

- sampling_protocol

4.919 `uninstall/0`

- `packs`

4.920 `uninstall/1`

- `packs`

4.921 `uninstall/2`

- `packs`

4.922 `union/3`

- `setp`

4.923 `union/4`

- `setp`
- `union_find_protocol`

4.924 `union_all/3`

- `union_find_protocol`

4.925 `unpin/0`

- `packs_common`

4.926 `unpin/1`

- `packs_common`

4.927 unzip/2

- zipperp

4.928 update/0

- doclet
- packs
- registries

4.929 update/1

- observer
- packs
- registries

4.930 update/2

- packs
- registries

4.931 update/3

- dictionaryp
- packs

4.932 update/4

- dictionaryp

4.933 update/5

- dictionaryp

4.934 update_in/4

- nested_dictionary_protocol

4.935 update_in/5

- nested_dictionary_protocol

4.936 uuid_null/1

- uuid_protocol

4.937 uuid_v1/2

- uuid_protocol

4.938 uuid_v4/1

- uuid_protocol

4.939 valid/1

- intervalp
- statisticsp
- term p
- varlistp

4.940 valid/2

- type

4.941 valid/3

- datep
- timep

4.942 valid_date/3

- iso8601

4.943 valid_option/1

- options_protocol

4.944 valid_options/1

- options_protocol

4.945 validate/1

- flags_validator

4.946 value/1

- counter

4.947 value/3

- pairs

4.948 value_reference/2

- java_utils_protocol

4.949 values/2

- dictionaryp
- pairs

4.950 variables/2

- term

4.951 variance/2

- statisticsp

4.952 variant/2

- lgtunit
- term

4.953 varnumbers/2

- term

4.954 varnumbers/3

- term

4.955 verify__commands__availability/0

- packs__common

4.956 version/6

- pack__protocol

4.957 versions/3

- packs

4.958 void/1

- java_utils_protocol

4.959 void_element/1

- html

4.960 von_mises/3

- sampling_protocol

4.961 wald/3

- sampling_protocol

4.962 wall_time/1

- osp

4.963 weibull/3

- sampling_protocol

4.964 weighted_mean/3

- statisticsp

4.965 welcome/0

- shell

4.966 when/2

- coroutining

4.967 whiledo/2

- looppp

4.968 white_space//0

- blank_grammars(Format)

4.969 white_spaces//0

- blank_grammars(Format)

4.970 with_output_to/2

- term_io_protocol

4.971 without//2

- sequence_grammars

4.972 working_directory/1

- osp

4.973 write__file/3

- csv_protocol
- tsv_protocol

4.974 write__max__depth/1

- debuggerp

4.975 write__stream/3

- csv_protocol
- tsv_protocol

4.976 write__term__to__atom/3

- term_io_protocol

4.977 write__term__to__chars/3

- term_io_protocol

4.978 write__term__to__chars/4

- term_io_protocol

4.979 write__term__to__codes/3

- term_io_protocol

4.980 write__term__to__codes/4

- term_io_protocol

4.981 write_to_atom/2

- term_io_protocol

4.982 write_to_chars/2

- term_io_protocol

4.983 write_to_codes/2

- term_io_protocol

4.984 z_normalization/2

- statisticsp

4.985 zero_or_more//0

- sequence_grammars

4.986 zero_or_more//1

- sequence_grammars

4.987 zero_or_more//2

- sequence_grammars

4.988 zip/2

- zipperp

4.989 zip/3

- zipperp

4.990 zip_at_index/4

- zlist

INDICES AND TABLES

- `genindex`
- `search`

Generated on Tue Nov 18 15:55:34 WET 2025

Symbols

(/)/2, 362
 (//)/2, 362
 (<)/2, 878
 (<=)/2, 15, 556
 (:=)/2, 879
 (=<)/2, 878
 (=>)/2, 16, 557
 (=\\=)/2, 879
 (>)/2, 878
 (>=)/2, 879
 =~= / 2, 502, 920

A

a_star_interpreter(W), 982
 absolute_file_name/2, 649
 acc_info/5, 274
 acc_info/7, 273
 activate_debug_handler/1, 75
 activate_monitor/0, 285
 active_debug_handler/1, 74
 active_debug_handler_/1, 82
 add/1, 717
 add/2, 716
 add/3, 715, 883
 add_directive_/2, 1024
 add_directive_/3, 1025
 add_directive_before_entity_/2, 1024
 add_library_documentation_url/4, 231
 add_link_options/3, 176
 add_node_zoom_option/4, 177
 addDependent/1, 153
 after/2, 420
 after/3, 83
 after_event_registry, 275
 all/0, 32, 118, 483
 all/1, 31, 118, 483
 all_files/0, 167, 190
 all_files/1, 166, 189
 all_libraries/0, 160, 183
 all_libraries/1, 160, 183
 all_score/1, 34

ancestor/1, 379
 ancestor/4, 42
 ancestors/1, 379
 apis/0, 368
 apis/1, 368
 append/2, 896
 append/3, 744, 896, 955
 apply/2, 1040
 apply/4, 263
 approximately_equal/2, 500, 918
 approximately_equal/3, 500, 918
 arbitrary, 1
 arbitrary/1, 4
 arbitrary/2, 4
 archive/1, 727
 arithmetic_mean/2, 821
 arithmetic_mean/5, 816
 array_list/2, 453
 array_to_list/2, 453
 array_to_terms/2, 452
 array_to_terms/3, 452
 as_curly_bracketed/2, 255, 616
 as_dictionary/2, 254
 as_difflist/2, 891
 as_heap/2, 356
 as_list/2, 254, 356, 745, 803, 883
 as_nested_dictionary/2, 615
 as_set/2, 803
 ask_question/5, 72
 assertion/1, 10, 495
 assertion/2, 10, 495
 assertions, 7
 assertions(Mode), 9
 assertions_messages, 11
 assignable/1, 14
 assignable/2, 15
 assignvars, 12
 assignvarsp, 13
 atom, 862
 atomic, 864
 automation_report, 484
 auxiliary_predicate_counter_/1, 533

available/0, 671
available/1, 670
available/2, 670
average/2, 926
average_deviation/3, 823
avltree, 248

B

backend_adapter_hook, 388
backend_random, 747
base64, 17
base64url, 19
base_/2, 327
before/2, 420
before/3, 83
before_event_registry, 276
bench_goal/1, 995
benchmark/2, 498
benchmark/3, 499
benchmark/4, 499
benchmark_generators, 984
benchmark_reified/3, 498
bernoulli/2, 771
best_first, 985
beta/3, 771
between/3, 758, 887
between/4, 885
bfs_interpreter, 987
binary_file_assertion/3, 529
binary_input_assertion/2, 513
binary_input_assertion/3, 513
binary_output_assertion/2, 524
binary_output_assertion/3, 523
binary_output_contents/1, 524
binary_output_contents/2, 524
binomial/3, 771
bintree, 250
bit//1, 343
bits//1, 343
blank//0, 337
blank_grammars(Format), 333
blanks//0, 337
body_pred/1, 594
body_pred_call/2, 597
branch/2, 330
built_in_directive/4, 363
built_in_flag/2, 313
built_in_method/4, 364
built_in_non_terminal/4, 364
built_in_predicate/4, 363
bup_interpreter, 988

C

c/1, 992

calendar_month/3, 435
call_with_timeout/2, 841
call_with_timeout/3, 842
callable, 865
cat/2, 621
cbor, 21
cbor(StringRepresentation), 22
cc_metric, 24
cdata_generation//1, 1033
change_directory/1, 652
changed/0, 152
changed/1, 153
character, 866
character_data_format/3, 1033
characterp, 867
chebyshev_distance/3, 928
chebyshev_norm/2, 927
check/1, 943, 963
check/2, 951
check/3, 951
check_binary_file/2, 528
check_binary_input/1, 512
check_binary_input/2, 512
check_binary_output/1, 523
check_binary_output/2, 522
check_option/1, 637
check_options/1, 637
check_text_file/2, 527
check_text_file/3, 526
check_text_input/1, 509
check_text_input/2, 508
check_text_output/1, 517
check_text_output/2, 516
check_text_output/3, 516
chi_squared/2, 769
chr_is/2, 844
chr_next_state/1, 847
chr_no_spy/1, 845
chr_nospy/0, 845
chr_notrace/0, 844
chr_option/2, 845
chr_option_allow_deep_guards/0, 847
chr_option_optimization_level/1, 846
chr_option_print_trace/0, 845
chr_option_show_history/0, 846
chr_option_show_id/0, 847
chr_option_show_stack/0, 846
chr_option_show_store/0, 846
chr_option_trace_interactive/0, 846
chr_rule_/1, 848
chr_spy/1, 845
chr_spy_point/1, 847
chr_trace/0, 844
circular_uniform_cartesian/3, 776

circular_uniform_polar/3, 776
 class/1, 372
 class_hierarchy, 370
 class_hierarchyp, 371
 classes/1, 372
 clause/5, 737
 clause_/5, 738
 clause_breakpoint_/2, 131
 clause_location/6, 737
 clause_location_/6, 738
 clean/0, 689, 721
 clean/1, 688, 721
 clean/2, 688
 clean_binary_input/0, 514
 clean_binary_output/0, 525
 clean_directory/1, 530
 clean_file/1, 529
 clean_text_input/0, 510
 clean_text_output/0, 520
 cleanup/0, 504
 clone/1, 551, 727
 clone/3, 255
 clone/4, 256
 cloning, 550
 closed_input_stream/2, 530
 closed_output_stream/2, 531
 code_metric, 25
 code_metrics, 38
 code_metrics_messages, 40
 code_metrics_utilities, 41
 coefficient_of_variation/2, 824
 command/2, 706
 command_line_arguments/1, 662
 commit_author/2, 330
 commit_date/2, 331
 commit_hash/2, 331
 commit_hash_abbreviated/2, 332
 commit_log/3, 332
 commit_message/2, 332
 comparingp, 877
 compile_aux_clauses/1, 79
 compile_predicate_heads/4, 79
 compile_predicate_indicators/3, 80
 compiled_pred_call/2, 596
 completion/2, 362
 completions/2, 363
 compound, 880
 condition/0, 504
 conditional_breakpoint_/3, 133
 connect/1, 796
 connect/3, 796
 console/1, 797
 contains/2, 423
 context_breakpoint_/4, 132

control//0, 338
 control_construct/4, 364
 controls//0, 339
 copy_file/2, 658
 core_messages, 63
 coroutining, 86
 counter, 989
 counter/2, 552, 609
 counter_/2, 324, 554, 610
 counters, 551
 coupling_metric, 46
 cover/1, 491
 coverage_report, 485
 covered_/4, 535
 cpu_time/1, 111, 660
 create_binary_file/2, 526
 create_text_file/2, 526
 create_text_file/3, 525
 csv, 89
 csv(Header,Separator,IgnoreQuotes), 90
 csv_guess_questions, 92
 csv_protocol, 93
 current/2, 1037
 current_entity/1, 43
 current_prog/1, 845

D

d2_graph_language, 155
 data/0, 735
 data/1, 735
 data/2, 735
 databasep, 993
 date, 104
 date/4, 427
 date/5, 428
 date/6, 428
 date/7, 430
 date_string/3, 431
 date_time/7, 660
 datep, 105
 days_in_month/3, 107
 deactivate_debug_handler/0, 75
 dead_code_scanner, 112
 dead_code_scanner_messages, 120
 debug/0, 138
 debug_expansion(Mode), 995
 debug_handler/1, 74
 debug_handler/3, 75
 debug_messages, 121
 debugger, 125
 debugger_messages, 135
 debuggerp, 136
 debugging/0, 139
 debugging/1, 139

debugging_/0, 127
decide/1, 307
decide/2, 307
declares_predicate/2, 43
decode_exception/2, 455
decode_exception/3, 455
decode_url_spaces/2, 709
decompile_predicate_heads/4, 80
decompile_predicate_indicators/4, 81
decompose_file_name/3, 649
decompose_file_name/4, 649
decrement_counter/1, 553
default_atom_mutations, 599
default_compound_mutations, 600
default_float_mutations, 602
default_integer_mutations, 603
default_list_mutations, 604
default_option/1, 638, 1020
default_options/1, 639, 1020
default_workflow_hook, 390
define_flag/1, 314
define_flag/2, 315
define_log_file/2, 566
defined/4, 714
defined_flag/6, 313
defined_flag_/6, 315
defines_predicate/2, 43
defines_predicate/3, 44
del_monitors/0, 282
del_monitors/4, 281
del_spy_points/4, 287
delete/0, 721
delete/1, 720
delete/2, 719
delete/3, 804, 896, 955
delete/4, 256, 354
delete_all_after/2, 1043
delete_all_after_and_unzip/2, 1043
delete_all_before/2, 1042
delete_all_before_and_unzip/2, 1043
delete_and_next/2, 1042
delete_and_previous/2, 1041
delete_and_unzip/2, 1042
delete_directory/1, 651
delete_directory_and_contents/1, 652
delete_directory_contents/1, 652
delete_file/1, 658
delete_in/4, 617
delete_matches/3, 897
delete_max/4, 262
delete_min/4, 261
demodb, 996
dependent_/1, 154
dependents/1, 153, 694
dependents/2, 693
dependents/3, 693
depth/2, 941
descendant/1, 380
descendant_class/1, 377
descendant_classes/1, 377
descendant_instance/1, 376
descendant_instances/1, 376
descendants/1, 381
describe/1, 678, 713
describe/2, 678
description/1, 665, 726
deterministic/1, 494
deterministic/2, 494
dfs_interpreter, 998
diagram(Format), 156
diagram_caption/3, 168
diagram_description/1, 167
diagram_name_suffix/1, 168
diagrams, 179
diagrams(Format), 180
dictionary, 252
dif, 267
dif/1, 87, 268
dif/2, 87, 268
difflist, 881
digit//1, 343
digits//1, 344
directories/1, 480, 1017
directories/2, 162, 186, 479, 1017
directories/3, 162, 185
directory/1, 29, 115, 165, 188, 481, 702, 1017
directory/2, 28, 115, 164, 187, 480, 702, 1016
directory/3, 164, 187
directory_dependency_diagram, 190
directory_dependency_diagram(Format), 192
directory_diagram(Format), 194
directory_entity_/4, 470
directory_exists/1, 655
directory_files/2, 654
directory_files/3, 655
directory_load_diagram, 197
directory_load_diagram(Format), 199
directory_score/2, 33
dirichlet/2, 775
disable/1, 122
disable/2, 123
disable_logging/1, 568
disconnect/1, 797
disjoint/2, 804
disjoint_sets/2, 976
dit_metric, 48
doc_goal/1, 270
doc_metric, 49

doclet, 269
 doctype/1, 410
 document_generation//2, 1032
 dot//1, 346
 dot_graph_language, 201
 dowhile/2, 571
 drop/3, 915
 dump_trace, 148
 during/2, 423
 dynamic_directive_/3, 1023

E

easter_day/3, 435
 edcg, 271
 edge/5, 172
 edge/6, 221
 edge_/5, 179
 edge_case/2, 5
 either, 289
 either/3, 302
 empty/1, 258, 355, 615, 623, 742, 805, 897, 956
 empty_map/1, 1030
 enable/1, 122
 enable/2, 123
 enable_logging/1, 568
 enabled/1, 123
 enabled/2, 124
 enabled_/1, 124
 enabled_/2, 125
 ensure_directory/1, 656
 ensure_file/1, 659
 entity/1, 27, 114, 246, 365
 entity/2, 246
 entity_calls/3, 44
 entity_defines_/2, 739
 entity_diagram, 202
 entity_diagram(Format), 203
 entity_info_pair_score_hook/3, 51
 entity_info_score_hook/2, 51
 entity_kind/2, 45
 entity_predicate_breakpoint_/4, 132
 entity_predicates_weights_hook/2, 51
 entity_prefix/2, 79
 entity_property/2, 45
 entity_score/2, 32
 entity_updates/3, 45
 enumerate/2, 761
 environment_variable/2, 659
 epsilon/1, 502
 equal/2, 424, 804
 erase/1, 793
 essentially_equal/3, 501, 919
 euclidean_distance/3, 928
 euclidean_norm/2, 927

event_registry, 277
 event_registryp, 279
 exception/4, 1032
 exclude/3, 579
 execution_context/7, 81
 expand_library_alias_paths, 288
 expand_library_path/2, 76
 expanding, 64
 expected, 292
 expected(Expected), 297
 expected/1, 300
 expecteds/2, 290
 explain//1, 861
 explicit_tracing_/0, 128
 exponential/2, 770
 extension/1, 384
 extensions/1, 385
 external_predicate_/1, 248

F

f/4, 986
 failed_/3, 534
 failed_test_reason//1, 537
 false/1, 449
 fast_random, 749
 fault/5, 1032
 fcube, 305
 fcube/0, 307
 file/1, 28, 114, 205, 482, 1019
 file/2, 28, 114, 205, 482, 1019
 file_being_advised_/4, 1025
 file_dependency_diagram, 207
 file_dependency_diagram(Format), 209
 file_diagram(Format), 211
 file_exists/1, 656
 file_footer/3, 219
 file_header/3, 219
 file_line_hit_count_/3, 134
 file_load_diagram, 214
 file_load_diagram(Format), 216
 file_modification_time/2, 656
 file_path/2, 505
 file_permission/2, 657
 file_score/2, 33
 file_size/2, 657
 file_to_bytes/2, 782
 file_to_bytes/3, 782
 file_to_chars/2, 781
 file_to_chars/3, 781
 file_to_codes/2, 780
 file_to_codes/3, 780
 file_to_terms/2, 781
 file_to_terms/3, 782
 file_type_extension/2, 78

- files/1, 166, 189, 481, 1018
- files/2, 165, 189, 481, 1018
- files/3, 165, 188
- filter/2, 629
- filter_external_file_extension/3, 176
- filter_file_extension/3, 175
- find/4, 975
- find/5, 975
- findall_member/4, 579
- findall_member/5, 580
- finished_by/2, 424
- finishes/2, 423
- fired_/3, 535
- fisher/3, 769
- fix_option/2, 641
- fix_options/2, 640
- flag_group_chk/1, 312
- flag_groups/1, 312
- flag_value_/2, 315
- flags, 309
- flags_validator, 316
- flaky_/1, 535
- flat_map/2, 301, 630
- flatten/2, 897, 956
- flatten_goals//1, 1000
- flatting, 999
- float, 884
- float//1, 345
- fold_left/4, 581
- fold_left_1/3, 581
- fold_right/4, 583
- fold_right_1/3, 583
- fordownto/3, 574
- fordownto/4, 574
- fordownto/5, 575
- foreach/3, 572
- foreach/4, 572
- format, 318
- format/2, 319
- format/3, 319
- format_entity_score//2, 35
- format_object/1, 167
- format_to_atom/3, 837
- format_to_chars/3, 838
- format_to_chars/4, 838
- format_to_codes/3, 839
- format_to_codes/4, 839
- forto/3, 573
- forto/4, 573
- forto/5, 573
- forward/1, 67
- forward/2, 1039
- forward/3, 1039
- forwarding, 66

- fractile/3, 827
- freeze/2, 87
- from_generator/2, 296, 625
- from_generator/3, 295, 624
- from_generator/4, 295
- from_goal/2, 295, 624
- from_goal/3, 294, 624
- from_goal/4, 294
- frozen/2, 88
- full_device_path/1, 654
- func_test/3, 595
- functional/0, 595

G

- gamma/3, 772
- generate/1, 416, 968
- generate/2, 18, 20, 23, 409, 461, 467, 968
- generate/8, 968
- generated_predicate_/1, 589
- generating_/0, 540
- genint, 320
- genint/2, 323
- genint_core, 321
- gensym, 324
- gensym/2, 327
- gensym_core, 325
- geometric/2, 770
- geometric_mean/2, 821
- get/1, 631
- get_field/2, 443
- get_flag_value/2, 310
- get_seed/1, 6, 753
- git, 328
- git_protocol, 329
- gnu/0, 306
- goal_expansion/2, 65
- grammar_rules_hook, 391
- graph_footer/5, 220
- graph_header/5, 220
- graph_language_protocol, 218
- graph_language_registry, 222
- ground/1, 941
- ground_entity_identifier/3, 175
- group_by_key/2, 937
- group_consecutive_by_key/2, 937
- group_sorted_by_key/2, 936
- guess_arity/2, 102
- guess_separator/2, 102
- gumbel/3, 775

H

- halstead_metric, 54
- halstead_metric(Stroud), 55
- hamming_distance/3, 898

[handbook/0](#), 368
[handbook/1](#), 368
[harmonic_mean/2](#), 821
[head/2](#), 742
[head_pred/1](#), 594
[heap\(Order\)](#), 351
[heapp](#), 352
[help](#), 360
[help/0](#), 362, 699
[help_info_support](#), 366
[heuristic_expansion\(Mode\)](#), 1000
[hex_digit//1](#), 344
[hex_digits//1](#), 344
[hierarchyp](#), 378
[home/1](#), 666, 727
[hook_pipeline\(Pipeline\)](#), 385
[hook_set\(Set\)](#), 387
[html](#), 408
[html5](#), 411
[hypergeometric/4](#), 770

I

[ibk/3](#), 594
[iddfs_interpreter\(Increment\)](#), 1002
[identity_hook](#), 392
[ids](#), 413
[ids\(Representation,Bytes\)](#), 414
[if_empty/1](#), 628
[if_expected/1](#), 299
[if_expected_or_else/2](#), 300
[if_present/1](#), 628
[if_present_or_else/2](#), 629
[if_unexpected/1](#), 299
[include/3](#), 579
[included_directory_/1](#), 196
[included_entity_/1](#), 206
[included_file_/1](#), 213
[included_library_/2](#), 232
[included_module_/1](#), 206
[included_predicate_/1](#), 247
[increase/1](#), 991
[increment/0](#), 990
[increment_counter/1](#), 552
[inheritance_diagram](#), 223
[inheritance_diagram\(Format\)](#), 224
[init/0](#), 1012
[init_log_file/2](#), 566
[inorder/2](#), 251
[insert/3](#), 805
[insert/4](#), 256, 354
[insert_after/3](#), 1041
[insert_all/3](#), 354, 805
[insert_before/3](#), 1040
[insert_in/4](#), 617
[install/1](#), 682
[install/2](#), 682
[install/3](#), 681
[install/4](#), 679
[installed/0](#), 673
[installed/1](#), 672
[installed/3](#), 672
[installed/4](#), 671
[instance/1](#), 373
[instance/2](#), 793
[instances/1](#), 373
[integer](#), 886
[integer//1](#), 345
[internal_os_path/2](#), 650
[interpreterp](#), 1003
[intersect/2](#), 806
[intersection/2](#), 259
[intersection/3](#), 259, 806
[intersection/4](#), 806
[interval](#), 416
[intervalp](#), 418
[invocation_number_/1](#), 129
[invoke/1](#), 444
[invoke/2](#), 445
[ip_grammars\(Format\)](#), 339
[ipv4//1](#), 340
[ipv6//1](#), 341
[is_absolute_file_name/1](#), 648
[is_alpha/1](#), 870
[is_alphanumeric/1](#), 869
[is_ascii/1](#), 869
[is_bin_digit/1](#), 870
[is_control/1](#), 874
[is_dec_digit/1](#), 871
[is_empty/0](#), 627
[is_end_of_line/1](#), 875
[is_expected/0](#), 298
[is_false/1](#), 450
[is_hex_digit/1](#), 871
[is_layout/1](#), 873
[is_letter/1](#), 870
[is_lower_case/1](#), 872
[is_newline/1](#), 875
[is_null/1](#), 451
[is_object/1](#), 451
[is_octal_digit/1](#), 871
[is_period/1](#), 874
[is_present/0](#), 627
[is_punctuation/1](#), 874
[is_quote/1](#), 873
[is_true/1](#), 450
[is_unexpected/0](#), 298
[is_upper_case/1](#), 872
[is_validator/1](#), 316

is_void/1, 450
is_vowel/1, 872
is_white_space/1, 873
iso8601, 425
issue_creator, 436
iterator_element/2, 454

J

java, 438
java(Reference), 439
java(Reference,ReturnValue), 440
java_access_protocol, 442
java_hook, 445
java_utils_protocol, 447
join/3, 742
join_all/3, 743
json, 456
json(ObjectRepresentation,PairRepresentation,StringRepresentation), 458
json(StringRepresentation), 457
json_lines, 462
json_lines(ObjectRepresentation,PairRepresentation,StringRepresentation), 464
json_lines(StringRepresentation), 463
json_lines_protocol, 465
json_protocol, 460
jump/3, 743
jump_all/3, 744
jump_all_block/3, 744
jump_to_invocation_number_/1, 130

K

key/2, 935
keys/2, 262, 935
keys_values/3, 934
keysort/2, 898
kurtosis/2, 825

L

language_object/2, 223
last/2, 898, 956
leaf/1, 380
leaf_class/1, 375
leaf_classes/1, 376
leaf_instance/1, 375
leaf_instances/1, 375
leap_year/1, 106, 434
leaping_/1, 129
learn/0, 598
learn/1, 598
learn/2, 592
learn/3, 592
learn_seq/2, 592

learn_with_timeout/4, 593
leash/1, 140
leashing/1, 141
leashing_/1, 129
least_common_multiple/2, 931
leaves/1, 380
length/2, 745, 899, 957
lgtdoc, 468
lgtdoc_messages, 471
lgtdocp, 472
lgtunit, 487
lgtunit_messages, 536
libraries/1, 160, 183, 476
libraries/2, 159, 182, 476
libraries/3, 159, 182
library/0, 365
library/1, 30, 117, 162, 185, 365, 477
library/2, 30, 116, 161, 184, 477
library_dependency_diagram, 226
library_dependency_diagram(Format), 227
library_diagram(Format), 229
library_entity, 469
library_load_diagram, 233
library_load_diagram(Format), 235
library_score/2, 32
license/1, 665
line_to_chars/2, 786
line_to_chars/3, 786
line_to_codes/2, 786
line_to_codes/3, 787
lint/0, 696, 723
lint/1, 695, 722
lint/2, 695
list, 890
list(Type), 892
list/0, 713
list_to_array/2, 453
listing, 558
listing/0, 559
listing/1, 559
listp, 893
load_registry/1, 707
loaded_file/1, 76
loaded_file_property/2, 76, 239
loaded_files_topological_sort/1, 77
loaded_files_topological_sort/2, 78
locate_directory/2, 174
locate_file/5, 175
locate_library/2, 174
log/3, 145
log_event/2, 567
log_file/2, 566
log_file_/2, 562, 564
log_point_/3, 131

logger, 561
 logging, 563
 logging/1, 567
 logging/3, 145
 logging_to_file_/2, 562, 564
 loggingp, 565
 logistic/3, 772
 lognormal/3, 768
 logseries/2, 769
 logtalk, 67
 logtalk_packs/0, 704
 logtalk_packs/1, 704
 lookup/2, 259
 lookup/3, 258
 lookup_in/3, 616
 loop, 569
 loopp, 570
 lower_upper/2, 876

M

magic, 1005
 magic/2, 1006
 magic_expansion(Mode), 1007
 magicise/4, 1006
 make/1, 505
 make_directory/1, 650
 make_directory_path/1, 651
 make_set/3, 974
 man/1, 369
 manhattan_distance/3, 929
 manhattan_norm/2, 928
 manuals/0, 366
 map/2, 263, 301, 584, 629, 746
 map/3, 263, 585, 746, 938
 map/4, 585
 map/5, 585
 map/6, 586
 map/7, 586
 map/8, 587
 map_element/2, 454
 map_member/3, 1030
 map_reduce/5, 587
 map_store/4, 1031
 max/2, 820, 899, 924
 max/3, 261
 max_clauses/1, 595
 max_inv_preds/1, 595
 max_size/1, 7
 maxheap, 357
 maybe, 620
 maybe/0, 763
 maybe/1, 764
 maybe/2, 764
 maybe_call/1, 764
 maybe_call/2, 765
 mean_deviation/2, 823
 median/2, 822, 926
 median_deviation/2, 824
 meets/2, 421
 member/2, 759, 807, 899
 memberchk/2, 807, 900, 957
 merge/3, 355
 merge_options/2, 640, 1021
 mermaid_graph_language, 237
 message_cache_/1, 544, 546, 547, 549
 message_diagram_description/1, 178
 message_hook/4, 72
 message_prefix_file/6, 71
 message_prefix_stream/4, 71
 message_tokens//2, 71
 met_by/2, 421
 meta, 576
 meta_compiler, 588
 meta_type/3, 950
 metagol, 590
 metagol_example_protocol, 597
 metap, 577
 metarule/6, 594
 metarule_next_id/1, 596
 min/2, 819, 900, 924
 min/3, 260
 min_clauses/1, 595
 min_max/3, 820, 925
 minheap, 359
 minimal_output, 538
 missing_predicate_directive_/3, 1023
 modes/2, 823, 926
 module_predicate_called_/3, 1022
 module_property/2, 239
 modules_diagram_support, 238
 monitor, 282
 monitor/1, 280
 monitor/4, 281
 monitor_activated/0, 285
 monitored/1, 280
 monitoring, 82
 monitorp, 284
 monitors/1, 280
 msort/2, 900
 msort/3, 901
 multifile_directive_/3, 1024
 mutation/3, 607, 609
 mutation/4, 610
 mutations, 606
 mutations_store, 608

N

name/1, 665, 726

name_of_day/3, 107
name_of_month/3, 107
natural, 916
natural//1, 345
navltree, 611
nbintree, 612
nested_dictionary_protocol, 613
new/1, 444, 556, 615, 942
new/2, 444, 556, 974
new/3, 419
new_line//0, 336
new_lines//0, 337
next/2, 1037
next/3, 1037
next/4, 260
nextto/3, 901, 957
noc_metric, 57
node/6, 171
node/7, 220
node_/6, 178
node_path_/2, 178
nodebug/0, 139
nolog/3, 146
nologall/0, 146
non_blank//1, 338
non_blanks//1, 338
non_standard_predicate_call_/2, 1023
nor_metric, 58
normal/3, 768
normal_element/2, 410
normalize_range/2, 929
normalize_range/4, 930
normalize_scalar/2, 931
normalize_unit/2, 930
nospy/1, 142
nospy/3, 143
nospy/4, 144
nospyall/0, 144
not_excluded_file/3, 46
not_excluded_file/4, 173
note/1, 505
note/2, 728
note/3, 667
notrace/0, 140
now/3, 111
nrmtree, 618
nth0/3, 902, 958
nth0/4, 902, 958
nth1/3, 902, 959
nth1/4, 903, 959
null/1, 449
null_device_path/1, 653
number, 917
number//1, 346

number_grammars(Format), 341
number_of_tests/1, 493
numberlist, 921
numberlistp, 922
numbervars/1, 945
numbervars/3, 945

O

object_file_/2, 487
object_predicate_called_/3, 1022
object_wrapper_hook, 394
object_wrapper_hook(Name,Relations), 397
object_wrapper_hook(Protocol), 395
observer, 150
occurrences/2, 904
occurrences/3, 904
occurs/2, 942
of/2, 623
of_expected/2, 293
of_unexpected/2, 293
omit_path_prefix/3, 177
one_or_more//0, 350
one_or_more//1, 349
one_or_more//2, 348
operating_system_machine/1, 662
operating_system_name/1, 661
operating_system_release/1, 662
operating_system_type/1, 661
option/2, 639
option/3, 639
optional, 622
optional(Optional), 626
options, 634
options_protocol, 635
or/2, 630
or_else/2, 302, 631
or_else_call/2, 303, 632
or_else_fail/1, 304, 633
or_else_get/2, 303, 632
or_else_throw/1, 304
or_else_throw/2, 633
orphaned/0, 677
orphaned/2, 676
os, 641
os_types, 643
osp, 645
outdated/0, 676
outdated/1, 676
outdated/2, 675
outdated/4, 674
outdated/5, 673
output_edges/1, 172
output externals/1, 169
output_file/4, 170

output_file_name/2, 219
 output_file_path/4, 174
 output_files/2, 170
 output_library/3, 169
 output_missing externals/1, 173
 output_node/6, 171
 output_rdirectory/3, 169
 output_rlibrary/3, 168
 output_sub_diagrams/1, 170
 overlapped_by/2, 422
 overlaps/2, 421

P

pack_protocol, 664
 packs, 668
 packs_common, 696
 packs_messages, 709
 packs_specs_hook, 710
 pairs, 933
 parent/1, 384
 parenthesis/2, 875
 parents/1, 384
 parse/2, 18, 20, 23, 461, 466, 1028
 parse/3, 1028
 parse_domain/2, 730
 parse_domain/3, 730
 parse_problem/2, 730
 parse_problem/3, 731
 partial_/1, 540, 542
 partial_map/4, 266
 partition/3, 291
 partition/4, 580, 801
 partition/5, 905
 partition/6, 581
 pass_info/1, 274
 pass_info/2, 274
 passed_/3, 534
 path_concat/3, 650
 pcd_data_7bit//1, 1033
 pddl, 729
 permutation/2, 761, 905, 959
 pid/1, 647
 pin/0, 700
 pin/1, 699
 pinned/1, 701
 plus/3, 888
 poisson/2, 772
 population, 811
 port/5, 736
 port_/5, 738
 portray_clause/1, 560
 ports_profiler, 733
 postorder/2, 252
 power/2, 773
 power_sequence/4, 889
 powerset/2, 808
 pp/1, 1029
 pp_string/1, 1031
 pprint/1, 593
 pprint_clause/1, 596
 pprint_clauses/1, 596
 pred_info/3, 273
 predicate/2, 119
 predicate_breakpoint_/3, 132
 predicate_called_but_not_defined_/2, 1021
 predicate_entity_/4, 470
 predicate_info_pair_score_hook/4, 53
 predicate_info_score_hook/3, 52
 predicate_mode_score_hook/3, 52
 predicate_mode_score_hook/5, 52
 predicates/2, 118
 prefix/0, 705
 prefix/1, 705
 prefix/2, 905, 960
 prefix/3, 906
 preorder/2, 251
 previous/2, 1038
 previous/3, 1038
 previous/4, 260
 print_flags/0, 312, 317
 print_flags/1, 313
 print_goal_hook, 398
 print_message/3, 70
 print_message_token/4, 70
 print_message_tokens/3, 70
 print_readme_file_path/1, 706
 process_all/1, 37
 process_directory/2, 36
 process_entity/2, 35
 process_file/2, 35
 process_library/2, 36
 process_rdirectory/2, 36
 process_rlibrary/2, 37
 product/2, 819, 925
 product/3, 808
 program_to_clauses/2, 593
 prolog_module_hook(Module), 400
 proper_prefix/2, 906
 proper_prefix/3, 907
 proper_suffix/2, 914
 proper_suffix/3, 914
 proto_hierarchy, 381
 proto_hierarchyp, 382
 prove/2, 1004
 prove/3, 1004
 provides/2, 722
 pseudo_random_protocol, 752

Q

quasi_skipping_/0, 128
question_hook/6, 73
question_prompt_stream/4, 73
queue, 739
queuep, 741
quick_check/1, 497
quick_check/2, 496
quick_check/3, 496

R

random, 754
random/1, 758
random/3, 762
random_node/1, 982
random_protocol, 757
random_tree/1, 985
randomize/1, 750, 755
randseq/4, 762
randset/4, 763
range/2, 820
rbtree, 265
rdirectories/1, 478
rdirectories/2, 478
rdirectory/1, 30, 116, 163, 187, 479, 1016
rdirectory/2, 29, 116, 163, 186, 478, 1016
rdirectory/3, 163, 186
rdirectory_score/2, 34
read_file, 731
read_file/2, 96, 732, 853
read_file/3, 94, 852
read_file_by_line/2, 99, 856
read_file_by_line/3, 97, 855
read_from_atom/2, 831
read_from_chars/2, 833
read_from_codes/2, 834
read_only_device_path/1, 654
read_stream/2, 96, 854
read_stream/3, 95, 852
read_stream_by_line/2, 99, 857
read_stream_by_line/3, 98, 856
read_term_from_atom/3, 831
read_term_from_chars/3, 832
read_term_from_chars/4, 832
read_term_from_codes/3, 833
read_term_from_codes/4, 833
reader, 779
readme/1, 703
readme/2, 703
readme_file_path/2, 706
record_/3, 794
recorda/2, 791
recorda/3, 790
recorded/2, 792

recorded/3, 792
recorded_database, 788
recorded_database_core, 789
recordz/2, 792
recordz/3, 791
redis, 795
reference_/1, 794
referenced_entity_/2, 206
referenced_logtalk_directory_/1, 197
referenced_logtalk_file_/1, 213
referenced_logtalk_library_/2, 232
referenced_module_/2, 207
referenced_predicate_/1, 247
referenced_prolog_directory_/1, 197
referenced_prolog_file_/1, 214
referenced_prolog_library_/2, 233
registries, 712
registry_loader_hook, 723
registry_protocol, 725
relative_standard_deviation/2, 825
remember_included_directory/1, 195
remember_included_file/1, 212
remember_included_library/2, 231
remember_referenced_logtalk_directory/1, 195
remember_referenced_logtalk_file/1, 212
remember_referenced_logtalk_library/2, 231
remember_referenced_prolog_directory/1, 196
remember_referenced_prolog_file/1, 213
remember_referenced_prolog_library/2, 232
remove_directive_/2, 1025
remove_duplicates/2, 907, 960
removeDependent/1, 154
rename_file/2, 658
replace/3, 1041
replace_sub_atom/4, 863
rescale/3, 931
reset/0, 138, 171, 698, 736, 992
reset/1, 736
reset_counter/1, 553
reset_counters/0, 553
reset_flags/0, 311
reset_flags/1, 311
reset_genint/0, 323
reset_genint/1, 323
reset_gensym/0, 326
reset_gensym/1, 326
reset_monitor/0, 286
reset_seed/0, 750, 755
restore/1, 692
restore/2, 691
reverse/2, 907, 960
rewind/2, 1038
rewind/3, 1039
rlibraries/1, 474

[rlibraries/2](#), 474
[rlibrary/1](#), 31, 117, 161, 184, 475
[rlibrary/2](#), 31, 117, 161, 184, 475
[rlibrary_score/2](#), 33
[rule/2](#), 994
[rule/3](#), 994
[rule/4](#), 994
[rule_expansion\(Mode\)](#), 1008
[run/0](#), 491
[run/1](#), 492
[run/2](#), 492
[run_quick_check_tests/5](#), 503
[run_test_set/0](#), 503
[run_test_sets/1](#), 493
[run_tests/0](#), 503
[run_tests/1](#), 503
[running_test_sets_/0](#), 532

S

[same_length/2](#), 908, 961
[same_length/3](#), 908
[sample](#), 813
[sampling_protocol](#), 766
[save/0](#), 1020
[save/1](#), 690, 1019
[save/2](#), 689
[save_edge/5](#), 172
[scalar_product/3](#), 929
[scan_left/4](#), 582
[scan_left_1/3](#), 582
[scan_right/4](#), 583
[scan_right_1/3](#), 584
[search/1](#), 679
[seed_/3](#), 751, 756
[select/3](#), 759, 808, 908, 961
[select/4](#), 760, 909
[selectchk/3](#), 809, 909
[selectchk/4](#), 909
[selected_test_/1](#), 533
[send/3](#), 797
[sequence/3](#), 888
[sequence/4](#), 761, 885, 889
[sequence/5](#), 885
[sequence_grammars](#), 347
[sequential_occurrences/2](#), 903
[sequential_occurrences/3](#), 903
[serve/3](#), 745
[set](#), 798
[set\(Type\)](#), 799
[set/1](#), 991
[set/4](#), 762
[set_binary_input/1](#), 511
[set_binary_input/2](#), 511
[set_binary_input/3](#), 510
[set_binary_output/1](#), 522
[set_binary_output/2](#), 521
[set_binary_output/3](#), 521
[set_element/2](#), 454
[set_field/2](#), 443
[set_flag_value/2](#), 310
[set_flag_value/3](#), 311
[set_monitor/4](#), 281
[set_seed/1](#), 6, 753
[set_spy_point/4](#), 287
[set_text_input/1](#), 508
[set_text_input/2](#), 507
[set_text_input/3](#), 507
[set_text_output/1](#), 515
[set_text_output/2](#), 515
[set_text_output/3](#), 514
[set_write_max_depth/1](#), 147
[setp](#), 802
[setup/0](#), 504, 698
[sha256sum_command/1](#), 707
[shell](#), 1009
[shell\(Interpreters\)](#), 1011
[shell/1](#), 648
[shell/2](#), 647
[shell_command/1](#), 270
[shell_expansion\(Mode\)](#), 1012
[shrink/3](#), 5
[shrink_sequence/3](#), 5
[shriner/1](#), 4
[sign//1](#), 346
[singletons/2](#), 945
[size/2](#), 264, 355, 807
[size_metric](#), 60
[skewness/2](#), 825
[skipped_/1](#), 534
[skipping_/0](#), 128
[skipping_unleashed_/1](#), 128
[sleep/1](#), 663
[softmax/2](#), 932
[softmax/3](#), 932
[sort/2](#), 801, 910
[sort/3](#), 910
[sort/4](#), 911
[source_file_extension/1](#), 240
[space//0](#), 335
[spaces//0](#), 335
[split/3](#), 863
[split/4](#), 911
[spy/1](#), 141
[spy/3](#), 142
[spy/4](#), 143
[spy_point/4](#), 286
[spy_point_/4](#), 284
[spying/1](#), 141

spying/3, 143
spying/4, 144
squares_and_cubes/6, 816
squares_and_hypers/6, 816
standard_cauchy/3, 777
standard_deviation/2, 824
standard_exponential/1, 777
standard_gamma/2, 777
standard_normal/1, 778
standard_t/2, 776
start/0, 734, 1010
start_redirect_to_file/2, 149
started_by/2, 422
starts/2, 422
statistics, 814
statisticsp, 817
stop/0, 734
stop_redirect_to_file/0, 149
stream_position/1, 531
stream_to_bytes/2, 785
stream_to_bytes/3, 785
stream_to_chars/2, 784
stream_to_chars/3, 784
stream_to_codes/2, 783
stream_to_codes/3, 783
stream_to_terms/2, 784
stream_to_terms/3, 785
streamvars, 554
sub_diagram_/1, 210, 217, 229, 236
sub_diagram_/2, 193, 200
sub_directory/2, 37
sub_library/2, 38
subclass/1, 373
subclasses/1, 374
subject, 151
sublist/2, 911, 961
subsequence/3, 912
subsequence/4, 912
subset/2, 809
substitute/4, 913
subsumes/2, 942
subterm/2, 943, 1029
subtract/3, 809, 913, 962
succ/2, 888
suffix/2, 913, 962
suffix/3, 914
sum/2, 819, 925
superclass/1, 374
superclasses/1, 374
supported_archive/1, 708
supported_editor_url_scheme_prefix/1, 176
supported_url_archive/1, 708
suppress_binary_output/0, 506
suppress_goal_hook, 401

suppress_text_output/0, 506
suspend_monitor/0, 286
swap/2, 760
swap_consecutive/2, 760
syndiff/3, 810

T

tab//0, 336
tabs//0, 336
take/3, 915
tap_output, 539
tap_report, 541
tar_command/1, 708
temporary_directory/1, 653
temporary_file_/1, 829
term, 938
term_expansion/2, 65
term_io, 828
term_io_protocol, 830
termp, 940
terms_to_array/2, 451
test/1, 493
test/2, 532
test/3, 532
test_/2, 533
test_count_/1, 541, 543
text_file_assertion/3, 528
text_file_assertion/4, 527
text_input_assertion/2, 510
text_input_assertion/3, 509
text_output_assertion/2, 518
text_output_assertion/3, 518
text_output_assertion/4, 517
text_output_contents/1, 520
text_output_contents/2, 519
text_output_contents/3, 519
time, 108
time_stamp/1, 659
timeout, 840
timeout/1, 596
timep, 110
timestamp/2, 969
timestamp/8, 969
timestamp_/6, 486
today/3, 106
tolerance_equal/4, 501, 919
top/3, 356
top_next/5, 357
toychrdb, 842
trace/0, 140
trace_event/2, 73
tracing_/0, 127
transpose/2, 936
triangular/4, 774

[triggered_breakpoint_/4](#), 133
[triggered_breakpoint_enabled_/2](#), 134
[true/1](#), 448
[tsv](#), 848
[tsv\(Header\)](#), 849
[tsv_protocol](#), 850
[tutor](#), 860
[type](#), 947
[type/1](#), 950
[type/3](#), 597
[type_entity_/4](#), 470

U

[ulid](#), 964
[ulid\(Representation\)](#), 965
[ulid_protocol](#), 966
[ulid_types](#), 970
[unexpected/1](#), 300
[unexpecteds/2](#), 291
[uniform/1](#), 774
[uniform/3](#), 773
[uninstall/0](#), 688
[uninstall/1](#), 687
[uninstall/2](#), 686
[union/3](#), 810
[union/4](#), 811, 974
[union_all/3](#), 975
[union_find](#), 971
[union_find_protocol](#), 972
[unknown_predicate_called_/2](#), 1022
[unpin/0](#), 701
[unpin/1](#), 700
[unsafe_set_flag_value/2](#), 314
[unzip/2](#), 1036
[update/0](#), 270, 686, 719
[update/1](#), 151, 685, 718
[update/2](#), 684, 717
[update/3](#), 257, 683
[update/4](#), 257
[update/5](#), 257
[update_in/4](#), 616
[update_in/5](#), 617
[upn_metric](#), 61
[user](#), 84
[uses_diagram](#), 241
[uses_diagram\(Format\)](#), 242
[uuid](#), 977
[uuid\(Representation\)](#), 978
[uuid_null/1](#), 981
[uuid_protocol](#), 979
[uuid_v1/2](#), 981
[uuid_v4/1](#), 981

V

[valid/1](#), 420, 827, 943, 962
[valid/2](#), 950
[valid/3](#), 108, 111
[valid_date/3](#), 433
[valid_option/1](#), 638
[valid_options/1](#), 638
[validate/1](#), 317
[validate/3](#), 316
[validate_type/1](#), 316
[value/1](#), 991
[value/3](#), 936
[value_reference/2](#), 448
[values/2](#), 262, 935
[variables/2](#), 944
[variance/2](#), 826
[variance/6](#), 817
[variant/2](#), 499, 944
[varlist](#), 952
[varlistp](#), 953
[varnumbers/2](#), 946
[varnumbers/3](#), 946
[verify_commands_availability/0](#), 699
[version/6](#), 666
[versions/3](#), 677
[void/1](#), 449
[void_element/1](#), 409
[von_mises/3](#), 775

W

[wald/3](#), 768
[wall_time/1](#), 660
[weibull/3](#), 773
[weighted_mean/3](#), 822
[welcome/0](#), 1010
[when/2](#), 88
[whiledo/2](#), 571
[white_space//0](#), 335
[white_spaces//0](#), 335
[with_output_to/2](#), 839
[without//2](#), 350
[working_directory/1](#), 653
[wrapper](#), 1014
[write_file/3](#), 100, 858
[write_max_depth/1](#), 146
[write_max_depth_/1](#), 130
[write_stream/3](#), 101, 858
[write_term_to_atom/3](#), 834
[write_term_to_chars/3](#), 835
[write_term_to_chars/4](#), 835
[write_term_to_codes/3](#), 836
[write_term_to_codes/4](#), 837
[write_to_atom/2](#), 835
[write_to_chars/2](#), 836

[write_to_codes/2](#), 837
[write_to_file_hook\(File\)](#), 402
[write_to_file_hook\(File,Options\)](#), 404
[write_to_stream_hook\(Stream\)](#), 405
[write_to_stream_hook\(Stream,Options\)](#), 406

X

[xhtml11](#), 412
[xml](#), 1026
[xml_to_document/3](#), 1030
[xref_diagram](#), 243
[xref_diagram\(Format\)](#), 245
[xunit_net_v2_output](#), 543
[xunit_net_v2_report](#), 545
[xunit_output](#), 546
[xunit_report](#), 548

Z

[z_normalization/2](#), 826
[zap_to_port_/1](#), 130
[zero_or_more//0](#), 350
[zero_or_more//1](#), 349
[zero_or_more//2](#), 348
[zip/2](#), 1036
[zip/3](#), 1036
[zip_at_index/4](#), 1045
[zipperp](#), 1034
[zlist](#), 1044