



CoreDX TM Data Distribution Service
The leading Small Footprint DDS Middleware

Java Reference Manual

Twin Oaks Computing, Inc
Castle Rock, CO 80108

Jan 2017



©2009-2017 Twin Oaks Computing, Inc

All rights reserved.

Published online 2009-2017

Trademarks

Twin Oaks Computing, and CoreDX DDS, and the CoreDX DDS logo are trademarks of Twin Oaks Computing, Inc. All other products or company names mentioned are used for identification purposes only, and may be trademarks of their respective owners.

Copy and Use Restrictions

No part of this document may be reproduced, stored, or transmitted (electronically or mechanically) without the prior written permission of Twin Oaks Computing, Inc. The software documented in this publication is provided pursuant to a License Agreement containing restrictions on its use.

DISCLAIMER OF WARRANTY

THIS DOCUMENT IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

Contact

Twin Oaks Computing, Inc
755 Maleta Ln, Ste 203
Castle Rock, CO 80108
(720) 733-7906
contact@twinoakscomputing.com
<http://www.twinoakscomputing.com>
http://twitter.com/CoreDX_DDS

Contents

1	Introduction	1
1.1	DDS Entities	4
1.1.1	Detailed Description	4
1.2	DDS Quality of Service	5
1.2.1	Detailed Description	6
1.2.2	Function Documentation	6
1.2.2.1	cleanup()	6
1.2.2.2	get_datareader_qos(String id)	6
1.2.2.3	get_datawriter_qos(String id)	7
1.2.2.4	get_participant_qos(String id)	7
1.2.2.5	get_participantfactory_qos(String id)	7
1.2.2.6	get_publisher_qos(String id)	7
1.2.2.7	get_subscriber_qos(String id)	7
1.2.2.8	get_topic_qos(String id)	7
1.2.2.9	qos_policy_str(int qos_policy_id)	7
1.2.2.10	QosProvider(String uri, String profile)	7
1.2.3	Variable Documentation	8
1.2.3.1	DATA_REPRESENTATION_QOS_POLICY_ID	8
1.2.3.2	DATAREADER_QOS_DEFAULT	8
1.2.3.3	DATAWRITER_QOS_DEFAULT	8
1.2.3.4	DEADLINE_QOS_POLICY_ID	8
1.2.3.5	DESTINATIONORDER_QOS_POLICY_ID	8
1.2.3.6	DURABILITY_QOS_POLICY_ID	8
1.2.3.7	DURABILITYSERVICE_QOS_POLICY_ID	8
1.2.3.8	ENTITYFACTORY_QOS_POLICY_ID	8
1.2.3.9	GROUPDATA_QOS_POLICY_ID	8
1.2.3.10	HISTORY_QOS_POLICY_ID	8
1.2.3.11	LATENCYBUDGET_QOS_POLICY_ID	8

1.2.3.12	LIFESPAN_QOS_POLICY_ID	8
1.2.3.13	LIVELINESS_QOS_POLICY_ID	8
1.2.3.14	OWNERSHIP_QOS_POLICY_ID	8
1.2.3.15	OWNERSHIPSTRENGTH_QOS_POLICY_ID	8
1.2.3.16	PARTICIPANT_QOS_DEFAULT	8
1.2.3.17	PARTITION_QOS_POLICY_ID	8
1.2.3.18	PRESENTATION_QOS_POLICY_ID	8
1.2.3.19	PUBLISHER_QOS_DEFAULT	8
1.2.3.20	READERDATA_LIFECYCLE_QOS_POLICY_ID	8
1.2.3.21	RELIABILITY_QOS_POLICY_ID	8
1.2.3.22	RESOURCELIMITS_QOS_POLICY_ID	8
1.2.3.23	SUBSCRIBER_QOS_DEFAULT	8
1.2.3.24	TIMEBASED_FILTER_QOS_POLICY_ID	8
1.2.3.25	TOPIC_QOS_DEFAULT	8
1.2.3.26	TOPICDATA_QOS_POLICY_ID	9
1.2.3.27	TRANSPORT_PRIORITY_QOS_POLICY_ID	9
1.2.3.28	USERDATA_QOS_POLICY_ID	9
1.2.3.29	WRITERDATA_LIFECYCLE_QOS_POLICY_ID	9
1.3	DDS Conditions, Listeners, and WaitSets	10
1.3.1	Detailed Description	10
1.3.2	Variable Documentation	10
1.3.2.1	ALIVE_INSTANCE_STATE	10
1.3.2.2	ANY_INSTANCE_STATE	10
1.3.2.3	ANY_SAMPLE_STATE	10
1.3.2.4	ANY_VIEW_STATE	10
1.3.2.5	NEW_VIEW_STATE	10
1.3.2.6	NOT_ALIVE_DISPOSED_INSTANCE_STATE	10
1.3.2.7	NOT_ALIVE_INSTANCE_STATE	10
1.3.2.8	NOT_ALIVE_NO_WRITERS_INSTANCE_STATE	10
1.3.2.9	NOT_NEW_VIEW_STATE	10
1.3.2.10	NOT_READ_SAMPLE_STATE	10
1.3.2.11	READ_SAMPLE_STATE	10
1.4	DDS Listeners	11
1.4.1	Detailed Description	11
1.5	DDS Conditions	12
1.5.1	Detailed Description	12
1.6	DDS WaitSets	13

1.6.1	Detailed Description	13
1.7	DDS Status Structures	14
1.7.1	Detailed Description	15
1.7.2	Variable Documentation	15
1.7.2.1	ALL_STATUS	15
1.7.2.2	DATA_AVAILABLE_STATUS	15
1.7.2.3	DATA_ON_READERS_STATUS	15
1.7.2.4	INCONSISTENT_TOPIC_STATUS	15
1.7.2.5	LIVELINESS_CHANGED_STATUS	15
1.7.2.6	LIVELINESS_LOST_STATUS	15
1.7.2.7	NO_STATUS	15
1.7.2.8	OFFERED_DEADLINE_MISSED_STATUS	15
1.7.2.9	OFFERED_INCOMPATIBLE_QOS_STATUS	15
1.7.2.10	PUBLICATION_MATCHED_STATUS	15
1.7.2.11	REQUESTED_DEADLINE_MISSED_STATUS	15
1.7.2.12	REQUESTED_INCOMPATIBLE_QOS_STATUS	15
1.7.2.13	SAMPLE_LOST_STATUS	15
1.7.2.14	SAMPLE_REJECTED_STATUS	15
1.7.2.15	STATUS_MASK_NONE	15
1.7.2.16	SUBSCRIPTION_MATCHED_STATUS	15
1.8	CoreDX DDS Transports	16
1.8.1	Detailed Description	16
1.9	CoreDX DDS RPC	17
1.9.1	Detailed Description	17
1.10	CoreDX DDS DynamicType	18
1.11	DDS Builtin Entities and Data Types	19
1.11.1	Detailed Description	19
2	Primary DDS Entities and Types	21
2.1	Condition Class Reference	21
2.1.1	Detailed Description	21
2.1.2	Member Function Documentation	21
2.1.2.1	get_trigger_value()	21
2.2	ContentFilteredTopic Class Reference	22
2.2.1	Detailed Description	22
2.2.2	Member Function Documentation	22
2.2.2.1	get_expression_parameters()	22

2.2.2.2	<code>get_related_topic()</code>	22
2.2.2.3	<code>set_expression_parameters(Vector filter_parameters)</code>	23
2.3	DataReader Class Reference	24
2.3.1	Detailed Description	24
2.3.2	Member Function Documentation	25
2.3.2.1	<code>create_querycondition(long sample_states, long view_states, long instance_states, String query_expression, Vector query_parameters)</code>	25
2.3.2.2	<code>create_readcondition(long sample_states, long view_states, long instance_states)</code>	25
2.3.2.3	<code>delete_contained_entities()</code>	25
2.3.2.4	<code>delete_readcondition(ReadCondition rc)</code>	26
2.3.2.5	<code>enable()</code>	26
2.3.2.6	<code>get_listener()</code>	26
2.3.2.7	<code>get_liveliness_changed_status(LivelinessChangedStatus status)</code>	26
2.3.2.8	<code>get_matched_publication_data(PublicationBuiltinTopicData publication_data, InstanceHandle_t publication_handle)</code>	26
2.3.2.9	<code>get_matched_publications(InstanceHandleSeq publication_handles)</code>	26
2.3.2.10	<code>get_qos(DataReaderQos qos)</code>	27
2.3.2.11	<code>get_requested_deadline_missed_status(RequestedDeadlineMissedStatus status)</code>	27
2.3.2.12	<code>get_requested_incompatible_qos_status(RequestedIncompatibleQosStatus status)</code>	27
2.3.2.13	<code>get_sample_lost_status(SampleLostStatus status)</code>	27
2.3.2.14	<code>get_sample_rejected_status(SampleRejectedStatus status)</code>	27
2.3.2.15	<code>get_subscriber()</code>	27
2.3.2.16	<code>get_subscription_matched_status(SubscriptionMatchedStatus status)</code>	27
2.3.2.17	<code>get_topicdescription()</code>	27
2.3.2.18	<code>set_listener(DataReaderListener new_listener, long mask)</code>	27
2.3.2.19	<code>set_qos(DataReaderQos qos)</code>	28
2.3.2.20	<code>wait_for_historical_data(Duration_t max_wait)</code>	28
2.4	DataWriter Class Reference	29
2.4.1	Detailed Description	29
2.4.2	Member Function Documentation	30
2.4.2.1	<code>assert_liveliness()</code>	30
2.4.2.2	<code>enable()</code>	30
2.4.2.3	<code>get_listener()</code>	30
2.4.2.4	<code>get_liveliness_lost_status(LivelinessLostStatus status)</code>	30
2.4.2.5	<code>get_matched_subscription_data(SubscriptionBuiltinTopicData subscription_data, InstanceHandle_t subscription_handle)</code>	30
2.4.2.6	<code>get_matched_subscriptions(InstanceHandleSeq subscription_handles)</code>	30
2.4.2.7	<code>get_offered_deadline_missed_status(OfferedDeadlineMissedStatus status)</code>	31

2.4.2.8	get_offered_incompatible_qos_status(OfferedIncompatibleQosStatus status)	31
2.4.2.9	get_publication_matched_status(PublicationMatchedStatus status)	31
2.4.2.10	get_publisher()	31
2.4.2.11	get_qos(DataWriterQos qos)	31
2.4.2.12	get_topic()	31
2.4.2.13	set_listener(DataWriterListener new_listener, long mask)	31
2.4.2.14	set_qos(DataWriterQos qos)	31
2.4.2.15	wait_for_acknowledgments(Duration_t max_wait)	32
2.5	DomainEntity Class Reference	33
2.5.1	Detailed Description	33
2.6	DomainParticipantFactory Class Reference	34
2.6.1	Detailed Description	34
2.6.2	Member Function Documentation	34
2.6.2.1	create_participant(long domain_id, DomainParticipantQos qos, DomainParticipantListener listener, long mask)	34
2.6.2.2	delete_participant(DomainParticipant participant)	35
2.6.2.3	get_default_participant_qos(DomainParticipantQos qos)	35
2.6.2.4	get_instance()	35
2.6.2.5	get_qos(DomainParticipantFactoryQos qos)	35
2.6.2.6	lookup_participant(long domain_id)	35
2.6.2.7	set_default_participant_qos(DomainParticipantQos qos)	35
2.6.2.8	set_qos(DomainParticipantFactoryQos qos)	36
2.7	DomainParticipant Class Reference	37
2.7.1	Detailed Description	38
2.7.2	Member Function Documentation	38
2.7.2.1	add_tcp_transport()	38
2.7.2.2	add_transport(Transport t)	38
2.7.2.3	assert_liveliness()	39
2.7.2.4	builtin_wait_for_acknowledgments(Duration_t max_wait)	39
2.7.2.5	contains_entity(InstanceHandle_t handle)	39
2.7.2.6	create_contentfilteredtopic(String name, Topic related_topic, String filter_expression, Vector filter_parameters)	39
2.7.2.7	create_multitopic(String name, String type_name, String subscription_expression, Vector expression_params)	39
2.7.2.8	create_publisher(PublisherQos qos, PublisherListener listener, long mask)	40
2.7.2.9	create_subscriber(SubscriberQos qos, SubscriberListener listener, long mask)	40
2.7.2.10	create_topic(String topic_name, String type_name, TopicQos qos, TopicListener listener, long mask)	40

2.7.2.11	<code>delete_contained_entities()</code>	40
2.7.2.12	<code>delete_contentfilteredtopic(ContentFilteredTopic cft)</code>	41
2.7.2.13	<code>delete_multitopic(MultiTopic a_multitopic)</code>	41
2.7.2.14	<code>delete_publisher(Publisher p)</code>	41
2.7.2.15	<code>delete_subscriber(Subscriber s)</code>	41
2.7.2.16	<code>delete_topic(Topic topic)</code>	41
2.7.2.17	<code>do_work(Duration_t time_slice)</code>	41
2.7.2.18	<code>enable()</code>	42
2.7.2.19	<code>find_topic(String topic_name, Duration_t timeout)</code>	42
2.7.2.20	<code>get_builtin_subscriber()</code>	42
2.7.2.21	<code>get_current_time(Time_t current_time)</code>	42
2.7.2.22	<code>get_default_publisher_qos(PublisherQos qos)</code>	43
2.7.2.23	<code>get_default_subscriber_qos(SubscriberQos qos)</code>	43
2.7.2.24	<code>get_default_topic_qos(TopicQos qos)</code>	43
2.7.2.25	<code>get_discovered_participant_data(ParticipantBuiltinTopicData participant_data, InstanceHandle_t participant_handle)</code>	43
2.7.2.26	<code>get_discovered_participants(InstanceHandleSeq participant_handles)</code>	43
2.7.2.27	<code>get_discovered_topic_data(TopicBuiltinTopicData topic_data, InstanceHandle_t topic_handle)</code>	43
2.7.2.28	<code>get_discovered_topics(Vector topic_handles)</code>	43
2.7.2.29	<code>get_domain_id()</code>	44
2.7.2.30	<code>get_listener()</code>	44
2.7.2.31	<code>get_qos(DomainParticipantQos qos)</code>	44
2.7.2.32	<code>ignore_participant(InstanceHandle_t handle)</code>	44
2.7.2.33	<code>ignore_publication(InstanceHandle_t handle)</code>	44
2.7.2.34	<code>ignore_subscription(InstanceHandle_t handle)</code>	44
2.7.2.35	<code>ignore_topic(InstanceHandle_t handle)</code>	44
2.7.2.36	<code>lookup_topicdescription(String name)</code>	45
2.7.2.37	<code>register_type(TypeSupport ts, String type_name)</code>	45
2.7.2.38	<code>set_default_publisher_qos(PublisherQos qos)</code>	45
2.7.2.39	<code>set_default_subscriber_qos(SubscriberQos qos)</code>	45
2.7.2.40	<code>set_default_topic_qos(TopicQos qos)</code>	45
2.7.2.41	<code>set_listener(DomainParticipantListener new_listener, long mask)</code>	45
2.7.2.42	<code>set_qos(DomainParticipantQos qos)</code>	46
2.8	Entity Class Reference	47
2.8.1	Detailed Description	47
2.8.2	Member Function Documentation	47

2.8.2.1	enable()	47
2.8.2.2	get_instance_handle()	47
2.8.2.3	get_status_changes()	47
2.8.2.4	get_statuscondition()	47
2.9	GuardCondition Class Reference	48
2.9.1	Detailed Description	48
2.9.2	Constructor & Destructor Documentation	48
2.9.2.1	GuardCondition()	48
2.10	Locator Class Reference	49
2.10.1	Detailed Description	49
2.10.2	Member Data Documentation	49
2.10.2.1	addr	49
2.11	MultiTopic Class Reference	50
2.11.1	Detailed Description	50
2.11.2	Member Function Documentation	50
2.11.2.1	get_name()	50
2.11.2.2	get_participant()	50
2.11.2.3	get_type_name()	50
2.12	ParticipantLocator Class Reference	51
2.12.1	Detailed Description	51
2.12.2	Member Data Documentation	51
2.12.2.1	participant_id	51
2.12.2.2	participant_id_max	51
2.12.2.3	participant_locator	51
2.13	Publisher Class Reference	52
2.13.1	Detailed Description	52
2.13.2	Member Function Documentation	52
2.13.2.1	begin_coherent_changes()	52
2.13.2.2	copy_from_topic_qos(DataWriterQos qos, TopicQos topic_qos)	53
2.13.2.3	create_datawriter(Topic topic, DataWriterQos qos, DataWriterListener listener, long mask)	53
2.13.2.4	delete_contained_entities()	53
2.13.2.5	delete_datawriter(DataWriter datawriter)	53
2.13.2.6	enable()	53
2.13.2.7	end_coherent_changes()	54
2.13.2.8	get_default_datawriter_qos(DataWriterQos qos)	54
2.13.2.9	get_listener()	54

2.13.2.10	get_participant()	54
2.13.2.11	get_qos(PublisherQos qos)	54
2.13.2.12	lookup_datawriter(String topic_name)	54
2.13.2.13	resume_publications()	54
2.13.2.14	set_default_datawriter_qos(DataWriterQos qos)	54
2.13.2.15	set_listener(PublisherListener new_listener, long mask)	54
2.13.2.16	set_qos(PublisherQos qos)	55
2.13.2.17	suspend_publications()	55
2.13.2.18	wait_for_acknowledgments(Duration_t max_wait)	55
2.14	QosPolicyCount Class Reference	56
2.14.1	Detailed Description	56
2.15	QueryCondition Class Reference	57
2.15.1	Detailed Description	57
2.15.2	Member Function Documentation	57
2.15.2.1	get_query_expression()	57
2.15.2.2	get_query_parameters()	58
2.15.2.3	set_query_parameters(Vector parameters)	58
2.16	ReadCondition Class Reference	59
2.16.1	Detailed Description	59
2.16.2	Member Function Documentation	59
2.16.2.1	get_datareader()	59
2.16.2.2	get_instance_state_mask()	59
2.16.2.3	get_sample_state_mask()	59
2.16.2.4	get_view_state_mask()	60
2.17	SampleInfo Class Reference	61
2.17.1	Detailed Description	61
2.17.2	Member Data Documentation	61
2.17.2.1	absolute_generation_rank	61
2.17.2.2	disposed_generation_count	61
2.17.2.3	generation_rank	61
2.17.2.4	instance_handle	62
2.17.2.5	instance_state	62
2.17.2.6	no_writers_generation_count	62
2.17.2.7	publication_handle	62
2.17.2.8	sample_rank	62
2.17.2.9	sample_state	62
2.17.2.10	source_timestamp	62

2.17.2.11	valid_data	62
2.17.2.12	view_state	63
2.18	StatusCondition Class Reference	64
2.18.1	Detailed Description	64
2.18.2	Member Function Documentation	64
2.18.2.1	get_enabled_statuses()	64
2.18.2.2	get_entity()	64
2.18.2.3	set_enabled_statuses(long mask)	64
2.19	Subscriber Class Reference	65
2.19.1	Detailed Description	65
2.19.2	Member Function Documentation	65
2.19.2.1	begin_access()	65
2.19.2.2	copy_from_topic_qos(DataReaderQos qos, TopicQos topic_qos)	66
2.19.2.3	create_datareader(TopicDescription topic, DataReaderQos qos, DataReaderListener listener, long mask)	66
2.19.2.4	delete_contained_entities()	66
2.19.2.5	delete_datareader(DataReader datareader)	66
2.19.2.6	enable()	66
2.19.2.7	end_access()	67
2.19.2.8	get_datareaders(Vector readers, long sample_states, long view_states, long instance_states)	67
2.19.2.9	get_default_datareader_qos(DataReaderQos qos)	67
2.19.2.10	get_listener()	67
2.19.2.11	get_participant()	67
2.19.2.12	get_qos(SubscriberQos qos)	67
2.19.2.13	lookup_datareader(String topic_name)	68
2.19.2.14	set_default_datareader_qos(DataReaderQos qos)	68
2.19.2.15	set_listener(SubscriberListener new_listener, long mask)	68
2.19.2.16	set_qos(SubscriberQos qos)	68
2.20	Topic Class Reference	69
2.20.1	Detailed Description	69
2.20.2	Member Function Documentation	69
2.20.2.1	enable()	69
2.20.2.2	get_inconsistent_topic_status(InconsistentTopicStatus status)	70
2.20.2.3	get_listener()	70
2.20.2.4	get_qos(TopicQos qos)	70
2.20.2.5	set_listener(TopicListener new_listener, long mask)	70

2.20.2.6	set_qos(TopicQos qos)	70
2.21	TopicDescription Interface Reference	71
2.21.1	Detailed Description	71
2.21.2	Member Function Documentation	71
2.21.2.1	get_name()	71
2.21.2.2	get_participant()	71
2.21.2.3	get_type_name()	71
2.22	TypeSupport Interface Reference	72
2.22.1	Detailed Description	72
2.22.2	Member Function Documentation	72
2.22.2.1	create_datareader(Subscriber sub, TopicDescription td, SWIGTYPE_p__DataReader jni_dr)	72
2.22.2.2	create_datawriter(Publisher pub, Topic topic, SWIGTYPE_p__DataWriter jni_dw)	72
2.22.2.3	get_fully_qualified_typename()	72
2.22.2.4	get_type()	72
2.22.2.5	get_typename()	73
2.22.2.6	getCTypeSupport()	73
2.22.2.7	register_type(DomainParticipant dp, String type_name)	73
2.23	WaitSet Class Reference	74
2.23.1	Detailed Description	74
2.23.2	Constructor & Destructor Documentation	74
2.23.2.1	WaitSet()	74
2.23.3	Member Function Documentation	74
2.23.3.1	attach_condition(Condition c)	74
2.23.3.2	destroy()	74
2.23.3.3	detach_condition(Condition c)	74
2.23.3.4	get_conditions(Vector attached_conditions)	75
2.23.3.5	wait(Vector active_conditions, Duration_t timeout)	75
3	Example DDS Generated Code	77
3.1	FooDataReader Class Reference	77
3.1.1	Detailed Description	78
3.1.2	Member Function Documentation	78
3.1.2.1	get_key_value(Foo key_holder, InstanceHandle_t handle)	78
3.1.2.2	lookup_instance(Foo instance_data)	78
3.1.2.3	read(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states)	78

3.1.2.4	read_instance(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t handle, long sample_states, long view_states, long instance_states)	79
3.1.2.5	read_next_instance(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t prev_handle, long sample_states, long view_states, long instance_states)	79
3.1.2.6	read_next_instance_w_condition(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t handle, ReadCondition condition)	80
3.1.2.7	read_next_sample(Foo received_data, SampleInfo sample_info)	80
3.1.2.8	read_w_condition(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, ReadCondition condition)	80
3.1.2.9	return_loan(FooSeq received_data, SampleInfoSeq sample_infos)	80
3.1.2.10	take(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states)	81
3.1.2.11	take_instance(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t handle, long sample_states, long view_states, long instance_states)	82
3.1.2.12	take_next_instance(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t prev_handle, long sample_states, long view_states, long instance_states)	82
3.1.2.13	take_next_instance_w_condition(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t handle, ReadCondition condition)	82
3.1.2.14	take_next_sample(Foo received_data, SampleInfo sample_info)	82
3.1.2.15	take_w_condition(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, ReadCondition condition)	83
3.2	FooDataWriter Class Reference	84
3.2.1	Detailed Description	84
3.2.2	Member Function Documentation	84
3.2.2.1	dispose(Foo instance_data, InstanceHandle_t handle)	84
3.2.2.2	dispose_w_timestamp(Foo instance_data, InstanceHandle_t handle, Time_t timestamp)	85
3.2.2.3	get_key_value(Foo key_holder, InstanceHandle_t handle)	85
3.2.2.4	lookup_instance(Foo instance_data)	85
3.2.2.5	register_instance(Foo instance_data)	85
3.2.2.6	register_instance_w_timestamp(Foo instance_data, Time_t ts)	85
3.2.2.7	unregister_instance(Foo instance_data, InstanceHandle_t handle)	85
3.2.2.8	unregister_instance_w_timestamp(Foo instance_data, InstanceHandle_t handle, Time_t timestamp)	85
3.2.2.9	write(Foo instance_data, InstanceHandle_t handle)	85
3.2.2.10	write_w_timestamp(Foo instance_data, InstanceHandle_t handle, Time_t timestamp)	86
3.3	FooTypeSupport Class Reference	87
3.3.1	Detailed Description	87
3.3.2	Member Function Documentation	87
3.3.2.1	get_type_name()	87

3.3.2.2	register_type(DomainParticipant dp, String type_name)	87
4	Entity QoS Collections	89
4.1	DataReaderQos Class Reference	89
4.1.1	Detailed Description	89
4.1.2	Member Data Documentation	90
4.1.2.1	deadline	90
4.1.2.2	destination_order	90
4.1.2.3	durability	90
4.1.2.4	entity_name	90
4.1.2.5	history	90
4.1.2.6	latency_budget	90
4.1.2.7	liveliness	90
4.1.2.8	logging	90
4.1.2.9	ownership	90
4.1.2.10	reader_data_lifecycle	91
4.1.2.11	reliability	91
4.1.2.12	representation	91
4.1.2.13	resource_limits	91
4.1.2.14	rpc	91
4.1.2.15	rtps_reader	91
4.1.2.16	time_based_filter	91
4.1.2.17	type_consistency	91
4.1.2.18	user_data	91
4.2	DataWriterQos Class Reference	92
4.2.1	Detailed Description	92
4.2.2	Member Data Documentation	92
4.2.2.1	deadline	92
4.2.2.2	destination_order	93
4.2.2.3	durability	93
4.2.2.4	durability_service	93
4.2.2.5	entity_name	93
4.2.2.6	history	93
4.2.2.7	latency_budget	93
4.2.2.8	lifespan	93
4.2.2.9	liveliness	93
4.2.2.10	logging	93

4.2.2.11	ownership	93
4.2.2.12	ownership_strength	93
4.2.2.13	reliability	94
4.2.2.14	representation	94
4.2.2.15	resource_limits	94
4.2.2.16	rpc	94
4.2.2.17	rtps_writer	94
4.2.2.18	transport_priority	94
4.2.2.19	user_data	94
4.2.2.20	writer_data_lifecycle	94
4.3	DomainParticipantFactoryQos Class Reference	95
4.3.1	Detailed Description	95
4.3.2	Member Data Documentation	95
4.3.2.1	entity_factory	95
4.4	DomainParticipantQos Class Reference	96
4.4.1	Detailed Description	96
4.4.2	Member Data Documentation	96
4.4.2.1	discovery	96
4.4.2.2	entity_factory	96
4.4.2.3	entity_name	97
4.4.2.4	logging	97
4.4.2.5	peer_participants	97
4.4.2.6	properties	97
4.4.2.7	thread_model	97
4.4.2.8	user_data	97
4.5	PublisherQos Class Reference	98
4.5.1	Detailed Description	98
4.5.2	Member Data Documentation	98
4.5.2.1	entity_factory	98
4.5.2.2	entity_name	98
4.5.2.3	group_data	98
4.5.2.4	logging	98
4.5.2.5	partition	99
4.5.2.6	presentation	99
4.6	SubscriberQos Class Reference	100
4.6.1	Detailed Description	100
4.6.2	Member Data Documentation	100

4.6.2.1	entity_factory	100
4.6.2.2	entity_name	100
4.6.2.3	group_data	100
4.6.2.4	logging	100
4.6.2.5	partition	101
4.6.2.6	presentation	101
4.7	TopicQos Class Reference	102
4.7.1	Detailed Description	102
4.7.2	Member Data Documentation	102
4.7.2.1	deadline	102
4.7.2.2	destination_order	102
4.7.2.3	durability	103
4.7.2.4	durability_service	103
4.7.2.5	entity_name	103
4.7.2.6	history	103
4.7.2.7	latency_budget	103
4.7.2.8	lifespan	103
4.7.2.9	liveliness	103
4.7.2.10	logging	103
4.7.2.11	ownership	103
4.7.2.12	reliability	103
4.7.2.13	resource_limits	103
4.7.2.14	topic_data	104
4.7.2.15	transport_priority	104
5	QoS Policies	105
5.1	DeadlineQosPolicy Class Reference	105
5.1.1	Detailed Description	105
5.2	DataRepresentationQosPolicy Class Reference	106
5.2.1	Detailed Description	106
5.2.2	Member Data Documentation	106
5.2.2.1	value	106
5.3	DestinationOrderQosPolicy Class Reference	107
5.3.1	Detailed Description	107
5.4	DiscoveryQosPolicy Class Reference	108
5.4.1	Detailed Description	108
5.5	DurabilityQosPolicy Class Reference	109

5.5.1	Detailed Description	109
5.6	DurabilityServiceQosPolicy Class Reference	110
5.6.1	Detailed Description	110
5.7	EntityFactoryQosPolicy Class Reference	111
5.7.1	Detailed Description	111
5.8	EntityNameQosPolicy Class Reference	112
5.8.1	Detailed Description	112
5.9	GroupDataQosPolicy Class Reference	113
5.9.1	Detailed Description	113
5.10	HistoryQosPolicy Class Reference	114
5.10.1	Detailed Description	114
5.11	LatencyBudgetQosPolicy Class Reference	115
5.11.1	Detailed Description	115
5.12	LifespanQosPolicy Class Reference	116
5.12.1	Detailed Description	116
5.13	LivelinessQosPolicy Class Reference	117
5.13.1	Detailed Description	117
5.14	LoggingQosPolicy Class Reference	118
5.14.1	Detailed Description	118
5.15	OwnershipQosPolicy Class Reference	119
5.15.1	Detailed Description	119
5.16	OwnershipStrengthQosPolicy Class Reference	120
5.16.1	Detailed Description	120
5.17	PartitionQosPolicy Class Reference	121
5.17.1	Detailed Description	121
5.18	PeerParticipantQosPolicy Class Reference	122
5.18.1	Detailed Description	122
5.18.2	Member Data Documentation	122
5.18.2.1	strict_match	122
5.18.2.2	value	122
5.19	PresentationQosPolicy Class Reference	123
5.19.1	Detailed Description	123
5.20	PropertyQosPolicy Class Reference	124
5.20.1	Detailed Description	124
5.20.2	Member Data Documentation	124
5.20.2.1	value	124
5.21	ReaderDataLifecycleQosPolicy Class Reference	125

5.21.1 Detailed Description	125
5.22 ReliabilityQosPolicy Class Reference	126
5.22.1 Detailed Description	126
5.22.2 Member Data Documentation	126
5.22.2.1 kind	126
5.22.2.2 max_blocking_time	126
5.23 ResourceLimitsQosPolicy Class Reference	127
5.23.1 Detailed Description	127
5.23.2 Member Data Documentation	127
5.23.2.1 preallocate_instances	127
5.23.2.2 preallocate_samples	127
5.24 RpcQosPolicy Class Reference	128
5.24.1 Detailed Description	128
5.24.2 Member Data Documentation	128
5.24.2.1 related_entity_guid	128
5.24.2.2 service_instance_name	128
5.24.2.3 topic_aliases	128
5.25 RTPSReaderQosPolicy Class Reference	129
5.25.1 Detailed Description	129
5.26 RTPSWriterQosPolicy Class Reference	130
5.26.1 Detailed Description	130
5.27 ThreadModelQosPolicy Class Reference	131
5.27.1 Detailed Description	131
5.28 TimeBasedFilterQosPolicy Class Reference	132
5.28.1 Detailed Description	132
5.29 TopicDataQosPolicy Class Reference	133
5.29.1 Detailed Description	133
5.30 TransportPriorityQosPolicy Class Reference	134
5.30.1 Detailed Description	134
5.31 TypecodeQosPolicy Class Reference	135
5.31.1 Detailed Description	135
5.32 TypeConsistencyEnforcementQosPolicy Class Reference	136
5.32.1 Detailed Description	136
5.32.2 Member Data Documentation	136
5.32.2.1 kind	136
5.33 UserDataQosPolicy Class Reference	137
5.33.1 Detailed Description	137

5.34	WriterDataLifecycleQosPolicy Class Reference	138
5.34.1	Detailed Description	138
5.35	DestinationOrderQosPolicyKind Enum Reference	139
5.35.1	Detailed Description	139
5.35.2	Member Data Documentation	139
5.35.2.1	BY_RECEPTION_TIMESTAMP_DESTINATIONORDER_QOS	139
5.36	DiscoveryQosPolicyDiscoveryKind Enum Reference	140
5.36.1	Detailed Description	140
5.36.2	Member Data Documentation	140
5.36.2.1	PEER_DISCOVERY_QOS	140
5.37	DurabilityQosPolicyKind Enum Reference	141
5.37.1	Detailed Description	141
5.37.2	Member Data Documentation	141
5.37.2.1	TRANSIENT_DURABILITY_QOS	141
5.37.2.2	TRANSIENT_LOCAL_DURABILITY_QOS	141
5.37.2.3	VOLATILE_DURABILITY_QOS	141
5.38	HistoryQosPolicyKind Enum Reference	142
5.38.1	Detailed Description	142
5.38.2	Member Data Documentation	142
5.38.2.1	KEEP_LAST_HISTORY_QOS	142
5.39	LivelinessQosPolicyKind Enum Reference	143
5.39.1	Detailed Description	143
5.39.2	Member Data Documentation	143
5.39.2.1	AUTOMATIC_LIVELINESS_QOS	143
5.39.2.2	MANUAL_BY_PARTICIPANT_LIVELINESS_QOS	143
5.40	OwnershipQosPolicyKind Enum Reference	144
5.40.1	Detailed Description	144
5.40.2	Member Data Documentation	144
5.40.2.1	SHARED_OWNERSHIP_QOS	144
5.41	PresentationQosPolicyAccessScopeKind Enum Reference	145
5.41.1	Detailed Description	145
5.41.2	Member Data Documentation	145
5.41.2.1	INSTANCE_PRESENTATION_QOS	145
5.41.2.2	TOPIC_PRESENTATION_QOS	145
5.42	ReliabilityQosPolicyKind Enum Reference	146
5.42.1	Detailed Description	146
5.43	TypeCodeKind Enum Reference	147

5.43.1 Detailed Description	147
5.43.2 Member Function Documentation	147
5.43.2.1 coreToEnum(DDS_TypeCodeKind k)	147
5.43.3 Member Data Documentation	147
5.43.3.1 TYPECODE_ALIAS	147
5.43.3.2 TYPECODE_ARRAY	148
5.43.3.3 TYPECODE_BOOLEAN	148
5.43.3.4 TYPECODE_CHAR	148
5.43.3.5 TYPECODE_DOUBLE	148
5.43.3.6 TYPECODE_ENUM	148
5.43.3.7 TYPECODE_FLOAT	148
5.43.3.8 TYPECODE_LONG	148
5.43.3.9 TYPECODE_LONGDOUBLE	148
5.43.3.10 TYPECODE_LONGLONG	148
5.43.3.11 TYPECODE_OCTET	148
5.43.3.12 TYPECODE_SEQUENCE	148
5.43.3.13 TYPECODE_SHORT	149
5.43.3.14 TYPECODE_STRING	149
5.43.3.15 TYPECODE_STRUCT	149
5.43.3.16 TYPECODE_ULONG	149
5.43.3.17 TYPECODE_ULONGLONG	149
5.43.3.18 TYPECODE_UNION	149
5.43.3.19 TYPECODE_USHORT	149
5.43.3.20 TYPECODE_VALUE	149
5.43.3.21 TYPECODE_WCHAR	149
5.43.3.22 TYPECODE_WSTRING	149
5.44 TypeConsistencyKind Enum Reference	150
5.44.1 Detailed Description	150
5.44.2 Member Data Documentation	150
5.44.2.1 ALLOW_TYPE_COERCION	150
5.44.2.2 DISALLOW_TYPE_COERCION	150
6 QoS Provider	151
6.1 QosProvider Class Reference	151
6.1.1 Detailed Description	151
7 DDS Listeners	153
7.1 DataReaderListener Interface Reference	153

7.1.1	Detailed Description	153
7.1.2	Member Function Documentation	153
7.1.2.1	get_nil_mask()	153
7.1.2.2	on_data_available(DataReader the_reader)	153
7.1.2.3	on_liveliness_changed(DataReader the_reader, LivelinessChangedStatus status)	154
7.1.2.4	on_requested_deadline_missed(DataReader the_reader, RequestedDeadlineMissedStatus status)	154
7.1.2.5	on_requested_incompatible_qos(DataReader the_reader, RequestedIncompatibleQosStatus status)	154
7.1.2.6	on_sample_lost(DataReader the_reader, SampleLostStatus status)	154
7.1.2.7	on_sample_rejected(DataReader the_reader, SampleRejectedStatus status)	154
7.1.2.8	on_subscription_matched(DataReader the_reader, SubscriptionMatchedStatus status)	154
7.2	DataWriterListener Interface Reference	155
7.2.1	Detailed Description	155
7.2.2	Member Function Documentation	155
7.2.2.1	get_nil_mask()	155
7.2.2.2	on_liveliness_lost(DataWriter writer, LivelinessLostStatus status)	155
7.2.2.3	on_offered_deadline_missed(DataWriter writer, OfferedDeadlineMissedStatus status)	155
7.2.2.4	on_offered_incompatible_qos(DataWriter writer, OfferedIncompatibleQosStatus status)	155
7.2.2.5	on_publication_matched(DataWriter writer, PublicationMatchedStatus status)	156
7.3	DomainParticipantListener Interface Reference	157
7.3.1	Detailed Description	157
7.3.2	Member Function Documentation	157
7.3.2.1	get_nil_mask()	157
7.3.2.2	on_data_available(DataReader the_reader)	157
7.3.2.3	on_data_on_readers(Subscriber the_subscriber)	157
7.3.2.4	on_inconsistent_topic(Topic the_topic, InconsistentTopicStatus status)	158
7.3.2.5	on_liveliness_changed(DataReader the_reader, LivelinessChangedStatus status)	158
7.3.2.6	on_liveliness_lost(DataWriter writer, LivelinessLostStatus status)	158
7.3.2.7	on_offered_deadline_missed(DataWriter writer, OfferedDeadlineMissedStatus status)	158
7.3.2.8	on_offered_incompatible_qos(DataWriter writer, OfferedIncompatibleQosStatus status)	158
7.3.2.9	on_publication_matched(DataWriter writer, PublicationMatchedStatus status)	158
7.3.2.10	on_requested_deadline_missed(DataReader the_reader, RequestedDeadlineMissedStatus status)	158
7.3.2.11	on_requested_incompatible_qos(DataReader the_reader, RequestedIncompatibleQosStatus status)	159
7.3.2.12	on_sample_lost(DataReader the_reader, SampleLostStatus status)	159
7.3.2.13	on_sample_rejected(DataReader the_reader, SampleRejectedStatus status)	159

7.3.2.14	<code>on_subscription_matched(DataReader the_reader, SubscriptionMatchedStatus status)</code>	159
7.4	PublisherListener Interface Reference	160
7.4.1	Detailed Description	160
7.4.2	Member Function Documentation	160
7.4.2.1	<code>get_nil_mask()</code>	160
7.4.2.2	<code>on_liveliness_lost(DataWriter writer, LivelinessLostStatus status)</code>	160
7.4.2.3	<code>on_offered_deadline_missed(DataWriter writer, OfferedDeadlineMissedStatus status)</code>	160
7.4.2.4	<code>on_offered_incompatible_qos(DataWriter writer, OfferedIncompatibleQosStatus status)</code>	160
7.4.2.5	<code>on_publication_matched(DataWriter writer, PublicationMatchedStatus status)</code>	160
7.5	SubscriberListener Interface Reference	162
7.5.1	Detailed Description	162
7.5.2	Member Function Documentation	162
7.5.2.1	<code>get_nil_mask()</code>	162
7.5.2.2	<code>on_data_available(DataReader the_reader)</code>	162
7.5.2.3	<code>on_data_on_readers(Subscriber the_subscriber)</code>	162
7.5.2.4	<code>on_liveliness_changed(DataReader the_reader, LivelinessChangedStatus status)</code>	162
7.5.2.5	<code>on_requested_deadline_missed(DataReader the_reader, RequestedDeadlineMissedStatus status)</code>	163
7.5.2.6	<code>on_requested_incompatible_qos(DataReader the_reader, RequestedIncompatibleQosStatus status)</code>	163
7.5.2.7	<code>on_sample_lost(DataReader the_reader, SampleLostStatus status)</code>	163
7.5.2.8	<code>on_sample_rejected(DataReader the_reader, SampleRejectedStatus status)</code>	163
7.5.2.9	<code>on_subscription_matched(DataReader the_reader, SubscriptionMatchedStatus status)</code>	163
7.6	TopicListener Interface Reference	164
7.6.1	Detailed Description	164
7.6.2	Member Function Documentation	164
7.6.2.1	<code>get_nil_mask()</code>	164
7.6.2.2	<code>on_inconsistent_topic(Topic the_topic, InconsistentTopicStatus status)</code>	164
8	DDS Status Structures	165
8.1	InconsistentTopicStatus Class Reference	165
8.1.1	Detailed Description	165
8.1.2	Member Function Documentation	165
8.1.2.1	<code>get_total_count()</code>	165
8.1.2.2	<code>get_total_count_change()</code>	165
8.2	LivelinessChangedStatus Class Reference	166
8.2.1	Detailed Description	166
8.2.2	Member Function Documentation	166

8.2.2.1	get_alive_count()	166
8.2.2.2	get_alive_count_change()	166
8.2.2.3	get_last_publication_handle()	166
8.2.2.4	get_not_alive_count()	166
8.2.2.5	get_not_alive_count_change()	166
8.3	LivelinessLostStatus Class Reference	167
8.3.1	Detailed Description	167
8.3.2	Member Function Documentation	167
8.3.2.1	get_total_count()	167
8.3.2.2	get_total_count_change()	167
8.4	OfferedDeadlineMissedStatus Class Reference	168
8.4.1	Detailed Description	168
8.4.2	Member Function Documentation	168
8.4.2.1	get_last_instance_handle()	168
8.4.2.2	get_total_count()	168
8.4.2.3	get_total_count_change()	168
8.5	OfferedIncompatibleQosStatus Class Reference	169
8.5.1	Detailed Description	169
8.5.2	Member Function Documentation	169
8.5.2.1	get_last_policy_id()	169
8.5.2.2	get_policies()	169
8.5.2.3	get_total_count()	169
8.5.2.4	get_total_count_change()	169
8.6	PublicationMatchedStatus Class Reference	170
8.6.1	Detailed Description	170
8.6.2	Member Function Documentation	170
8.6.2.1	get_current_count()	170
8.6.2.2	get_current_count_change()	170
8.6.2.3	get_last_subscription_handle()	170
8.6.2.4	get_total_count()	170
8.6.2.5	get_total_count_change()	170
8.7	RequestedDeadlineMissedStatus Class Reference	171
8.7.1	Detailed Description	171
8.7.2	Member Function Documentation	171
8.7.2.1	get_last_instance_handle()	171
8.7.2.2	get_total_count()	171
8.7.2.3	get_total_count_change()	171

8.8	RequestedIncompatibleQosStatus Class Reference	172
8.8.1	Detailed Description	172
8.8.2	Member Function Documentation	172
8.8.2.1	get_last_policy_id()	172
8.8.2.2	get_policies()	172
8.8.2.3	get_total_count()	172
8.8.2.4	get_total_count_change()	172
8.9	SampleLostStatus Class Reference	173
8.9.1	Detailed Description	173
8.9.2	Member Function Documentation	173
8.9.2.1	get_total_count()	173
8.9.2.2	get_total_count_change()	173
8.10	SampleRejectedStatus Class Reference	174
8.10.1	Detailed Description	174
8.10.2	Member Function Documentation	174
8.10.2.1	get_last_instance_handle()	174
8.10.2.2	get_last_reason()	174
8.10.2.3	get_total_count()	174
8.10.2.4	get_total_count_change()	174
8.11	SubscriptionMatchedException Class Reference	175
8.11.1	Detailed Description	175
8.11.2	Member Function Documentation	175
8.11.2.1	get_current_count()	175
8.11.2.2	get_current_count_change()	175
8.11.2.3	get_last_publication_handle()	175
8.11.2.4	get_total_count()	175
8.11.2.5	get_total_count_change()	175
9	DDS Samples	177
9.1	LoanedSamples< T > Class Template Reference	177
9.1.1	Detailed Description	177
9.1.2	Member Function Documentation	177
9.1.2.1	length()	177
9.1.2.2	release(DataReaderI< T > reader, SampleSeq< T > data_seq, SampleInfoSeq info_seq)	177
9.1.2.3	return_loan()	177
9.1.2.4	size()	178

10 Builtin DDS Types	179
10.1 DomainId_t Class Reference	179
10.1.1 Detailed Description	179
10.2 Duration_t Class Reference	180
10.2.1 Detailed Description	180
10.3 InstanceHandle_t Class Reference	181
10.3.1 Detailed Description	181
10.4 Property_t Class Reference	182
10.4.1 Detailed Description	182
10.4.2 Member Data Documentation	182
10.4.2.1 name	182
10.4.2.2 propagate	182
10.4.2.3 value	182
10.5 Time_t Class Reference	183
10.5.1 Detailed Description	183
11 DDS Discovery Types	185
11.1 ParticipantBuiltinTopicDataDataReader Class Reference	185
11.1.1 Detailed Description	185
11.2 ParticipantBuiltinTopicData Class Reference	186
11.2.1 Detailed Description	186
11.3 PublicationBuiltinTopicDataDataReader Class Reference	187
11.3.1 Detailed Description	187
11.4 PublicationBuiltinTopicData Class Reference	188
11.4.1 Detailed Description	188
11.5 SubscriptionBuiltinTopicDataDataReader Class Reference	189
11.5.1 Detailed Description	189
11.6 SubscriptionBuiltinTopicData Class Reference	190
11.6.1 Detailed Description	190
12 Supporting Types	191
12.1 Constants Class Reference	191
12.1.1 Detailed Description	192
13 X-Types DynamicType API	193
13.1 ArrayDynamicType Class Reference	193
13.1.1 Detailed Description	193
13.2 BooleanDynamicType Class Reference	194

13.2.1 Detailed Description	194
13.3 CharDynamicType Class Reference	195
13.3.1 Detailed Description	195
13.4 DoubleDynamicType Class Reference	196
13.4.1 Detailed Description	196
13.5 DynamicTypeDataReader Class Reference	197
13.5.1 Detailed Description	197
13.6 DynamicTypeDataWriter Class Reference	198
13.6.1 Detailed Description	198
13.7 DynamicType Class Reference	199
13.7.1 Detailed Description	199
13.7.2 Member Function Documentation	199
13.7.2.1 release()	199
13.8 DynamicTypeTypeSupport Class Reference	200
13.8.1 Detailed Description	200
13.8.2 Member Function Documentation	200
13.8.2.1 create_datareader(Subscriber sub, TopicDescription td, SWIGTYPE_p__DataReader jni_dr)	200
13.8.2.2 create_datawriter(Publisher pub, Topic topic, SWIGTYPE_p__DataWriter jni_dw)	200
13.8.2.3 getCTypeSupport()	200
13.8.2.4 register_type(DomainParticipant dp, String type_name)	201
13.9 EnumDynamicType Class Reference	202
13.9.1 Detailed Description	202
13.10FloatDynamicType Class Reference	203
13.10.1 Detailed Description	203
13.11LongDynamicType Class Reference	204
13.11.1 Detailed Description	204
13.12LongLongDynamicType Class Reference	205
13.12.1 Detailed Description	205
13.13OctetDynamicType Class Reference	206
13.13.1 Detailed Description	206
13.14SequenceDynamicType Class Reference	207
13.14.1 Detailed Description	207
13.15ShortDynamicType Class Reference	208
13.15.1 Detailed Description	208
13.16StringDynamicType Class Reference	209
13.16.1 Detailed Description	209

13.17StructDynamicType Class Reference	210
13.17.1 Detailed Description	210
13.18ULongDynamicType Class Reference	211
13.18.1 Detailed Description	211
13.19ULongLongDynamicType Class Reference	212
13.19.1 Detailed Description	212
13.20UnionDynamicType Class Reference	213
13.20.1 Detailed Description	213
13.21UShortDynamicType Class Reference	214
13.21.1 Detailed Description	214
14 CoreDX DDS Transports	215
14.1 IpTransportInterface Class Reference	215
14.1.1 Detailed Description	215
14.2 LmtTransportConfig Class Reference	216
14.2.1 Detailed Description	216
14.2.2 Member Function Documentation	216
14.2.2.1 get_default_config()	216
14.2.2.2 get_env_config()	216
14.2.3 Member Data Documentation	216
14.2.3.1 debug_flags	216
14.2.3.2 max_rx_buf_size	216
14.2.3.3 max_tx_size	217
14.2.3.4 so_rcvbuf	217
14.2.3.5 so_sndbuf	217
14.3 LmtTransport Class Reference	218
14.3.1 Detailed Description	218
14.4 TcpTransportConfig Class Reference	219
14.4.1 Detailed Description	219
14.4.2 Member Function Documentation	219
14.4.2.1 get_default_config()	219
14.4.2.2 get_env_config()	219
14.4.3 Member Data Documentation	219
14.4.3.1 dynamic_interfaces	219
14.4.3.2 interfaces	219
14.4.3.3 participant_index	220
14.4.3.4 tx_max_packet_size	220

14.5	TcpTransport Class Reference	221
14.5.1	Detailed Description	221
14.6	Transport Class Reference	222
14.6.1	Detailed Description	222
14.7	UdpTransportConfig Class Reference	223
14.7.1	Detailed Description	223
14.7.2	Member Function Documentation	223
14.7.2.1	get_default_config()	223
14.7.2.2	get_env_config()	224
14.7.3	Member Data Documentation	224
14.7.3.1	advertise_meta_multicast	224
14.7.3.2	advertise_user_multicast	224
14.7.3.3	broadcast_address	224
14.7.3.4	do_meta_broadcast	224
14.7.3.5	dynamic_interfaces	224
14.7.3.6	interfaces	224
14.7.3.7	meta_multicast_address_v4	224
14.7.3.8	meta_multicast_address_v6	224
14.7.3.9	multicast_ttl	224
14.7.3.10	participant_index	224
14.7.3.11	rx_init_buffer_size	225
14.7.3.12	rx_max_buffer_size	225
14.7.3.13	rx_meta_multicast	225
14.7.3.14	rx_user_multicast	225
14.7.3.15	so_rcvbuf	225
14.7.3.16	so_sndbuf	225
14.7.3.17	tx_max_packet_size	225
14.7.3.18	tx_meta_multicast	225
14.7.3.19	tx_meta_unicast	225
14.7.3.20	use_ipv4	225
14.7.3.21	use_ipv6	225
14.7.3.22	user_multicast_address_v4	226
14.7.3.23	user_multicast_address_v6	226
14.8	UdpTransport Class Reference	227
14.8.1	Detailed Description	227

15.1	ClientEndpoint< TReq extends RequestType, TRep extends ReplyType > Class Template Reference	229
15.1.1	Detailed Description	229
15.1.2	Constructor & Destructor Documentation	229
15.1.2.1	ClientEndpoint(Class< TReq > req_class, Class< TRep > rep_class, ClientParams cparams)	229
15.1.3	Member Function Documentation	230
15.1.3.1	get_client_params()	230
15.1.3.2	receive_reply(Sample< TRep > reply, SampleIdentity_t relatedRequestId)	230
15.1.3.3	send_request(TReq request)	230
15.2	ClientParams Class Reference	231
15.2.1	Detailed Description	231
15.2.2	Constructor & Destructor Documentation	231
15.2.2.1	ClientParams()	231
15.2.2.2	ClientParams(ClientParams other)	231
15.2.3	Member Function Documentation	231
15.2.3.1	datareader_qos(DataReaderQos qos)	231
15.2.3.2	datareader_qos()	232
15.2.3.3	datawriter_qos(DataWriterQos qos)	232
15.2.3.4	datawriter_qos()	232
15.2.3.5	domain_participant(DomainParticipant part)	232
15.2.3.6	domain_participant()	232
15.2.3.7	instance_name(String instance_name)	232
15.2.3.8	instance_name()	232
15.2.3.9	publisher(Publisher publisher)	232
15.2.3.10	publisher()	232
15.2.3.11	reply_topic_name(String rep_topic)	232
15.2.3.12	reply_topic_name()	232
15.2.3.13	request_topic_name(String req_topic)	233
15.2.3.14	request_topic_name()	233
15.2.3.15	service_name(String service_name)	233
15.2.3.16	service_name()	233
15.2.3.17	subscriber(Subscriber subscriber)	233
15.2.3.18	subscriber()	233
15.3	ReplierParams Class Reference	234
15.3.1	Detailed Description	234
15.3.2	Constructor & Destructor Documentation	234
15.3.2.1	ReplierParams()	234

15.3.2.2	ReplierParams(ReplierParams other)	234
15.3.3	Member Function Documentation	235
15.3.3.1	datareader_qos(DataReaderQos qos)	235
15.3.3.2	datareader_qos()	235
15.3.3.3	datawriter_qos(DataWriterQos qos)	235
15.3.3.4	datawriter_qos()	235
15.3.3.5	domain_participant(DomainParticipant participant)	235
15.3.3.6	domain_participant()	235
15.3.3.7	instance_name(String instance_name)	235
15.3.3.8	instance_name()	235
15.3.3.9	publisher(Publisher publisher)	235
15.3.3.10	publisher()	235
15.3.3.11	replier_listener(ReplierListener<?extends RequestType,?extends ReplyType > listener)	235
15.3.3.12	replier_listener()	236
15.3.3.13	reply_topic_name(String rep_topic)	236
15.3.3.14	reply_topic_name()	236
15.3.3.15	request_topic_name(String req_topic)	236
15.3.3.16	request_topic_name()	236
15.3.3.17	service_name(String service_name)	236
15.3.3.18	service_name()	236
15.3.3.19	simple_replier_listener(SimpleReplierListener<?extends RequestType,?extends ReplyType > listener)	236
15.3.3.20	simple_replier_listener()	236
15.3.3.21	subscriber(Subscriber subscriber)	236
15.3.3.22	subscriber()	236
15.4	Replier< TReq extends RequestType, TRep extends ReplyType > Class Template Reference	237
15.4.1	Detailed Description	237
15.4.2	Constructor & Destructor Documentation	237
15.4.2.1	Replier(Class< TReq > treq_clazz, Class< TRep > trep_clazz, ReplierParams r_params)	237
15.4.3	Member Function Documentation	238
15.4.3.1	close()	238
15.4.3.2	get_replier_params()	238
15.4.3.3	get_reply_datawriter()	238
15.4.3.4	get_request_datareader()	238
15.4.3.5	is_null()	238
15.4.3.6	read_request(Sample< TReq > request)	238

15.4.3.7	read_requests(int max_samples)	238
15.4.3.8	receive_nondata_samples(boolean enable)	239
15.4.3.9	receive_request(Sample< TReq > request, Duration_t max_wait)	239
15.4.3.10	receive_requests(Duration_t max_wait)	239
15.4.3.11	receive_requests(int min_request_count, int max_request_count, Duration_t max_wait)	239
15.4.3.12	send_reply(TRep reply, SampleIdentity_t related_request_id)	239
15.4.3.13	take_request(Sample< TReq > request)	239
15.4.3.14	take_requests(int max_samples)	239
15.4.3.15	wait_for_requests(Duration_t max_wait)	240
15.4.3.16	wait_for_requests(int min_count, Duration_t max_wait)	240
15.5	RequesterParams Class Reference	241
15.5.1	Detailed Description	241
15.5.2	Constructor & Destructor Documentation	241
15.5.2.1	RequesterParams()	241
15.5.2.2	RequesterParams(RequesterParams other)	241
15.5.3	Member Function Documentation	242
15.5.3.1	datareader_qos(DataReaderQos qos)	242
15.5.3.2	datareader_qos()	242
15.5.3.3	datawriter_qos(DataWriterQos qos)	242
15.5.3.4	datawriter_qos()	242
15.5.3.5	domain_participant(DomainParticipant participant)	242
15.5.3.6	domain_participant()	242
15.5.3.7	publisher(Publisher publisher)	242
15.5.3.8	publisher()	242
15.5.3.9	reply_topic_name(String name)	242
15.5.3.10	reply_topic_name()	243
15.5.3.11	request_topic_name(String name)	243
15.5.3.12	request_topic_name()	243
15.5.3.13	requester_listener(RequesterListener<?extends RequestType,?extends ReplyType > listener)	243
15.5.3.14	requester_listener()	243
15.5.3.15	service_name(String name)	243
15.5.3.16	service_name()	243
15.5.3.17	simple_requester_listener(SimpleRequesterListener<?extends ReplyType > listener)	243
15.5.3.18	simple_requester_listener()	243
15.5.3.19	subscriber(Subscriber subscriber)	244
15.5.3.20	subscriber()	244

15.6 Requester< TReq extends RequestType, TRep extends ReplyType > Class Template Reference	245
15.6.1 Detailed Description	245
15.6.2 Constructor & Destructor Documentation	246
15.6.2.1 Requester(Class< TReq > req_class, Class< TRep > rep_class)	246
15.6.2.2 Requester(Class< TReq > req_class, Class< TRep > rep_class, Requester-Params req_params)	246
15.6.3 Member Function Documentation	246
15.6.3.1 get_reply_datareader()	246
15.6.3.2 get_request_datawriter()	246
15.6.3.3 get_requester_params()	246
15.6.3.4 read_replies(int max_count)	246
15.6.3.5 read_replies(int max_count, SampleIdentity_t related_request_id)	246
15.6.3.6 read_replies(SampleIdentity_t related_request_id)	246
15.6.3.7 read_reply(Sample< TRep > reply)	247
15.6.3.8 read_reply(Sample< TRep > reply, SampleIdentity_t related_request_id)	247
15.6.3.9 receive_nondata_samples(boolean enable)	247
15.6.3.10 receive_replies(Duration_t max_wait)	247
15.6.3.11 receive_replies(int min_count, int max_count, Duration_t max_wait)	247
15.6.3.12 receive_reply(Sample< TRep > reply, Duration_t timeout)	247
15.6.3.13 receive_reply(Sample< TRep > reply, SampleIdentity_t relatedRequestId)	248
15.6.3.14 send_request(TReq request)	248
15.6.3.15 send_request_oneway(TReq request)	248
15.6.3.16 take_replies(int max_count)	248
15.6.3.17 take_replies(int max_count, SampleIdentity_t related_request_id)	248
15.6.3.18 take_replies(SampleIdentity_t related_request_id)	248
15.6.3.19 take_reply(Sample< TRep > reply)	248
15.6.3.20 take_reply(Sample< TRep > reply, SampleIdentity_t related_request_id)	249
15.6.3.21 wait_for_replies(int min_count, Duration_t max_wait)	249
15.6.3.22 wait_for_replies(Duration_t max_wait)	249
15.6.3.23 wait_for_replies(int min_count, Duration_t max_wait, SampleIdentity_t related_request_id)	249
15.7 ServiceEndpoint Class Reference	251
15.7.1 Detailed Description	251
15.7.2 Constructor & Destructor Documentation	251
15.7.2.1 ServiceEndpoint()	251
15.7.3 Member Function Documentation	251
15.7.3.1 close()	251
15.7.3.2 get_service_params()	251

15.7.3.3	is_null()	252
15.7.3.4	pause()	252
15.7.3.5	resume()	252
15.7.3.6	status()	252
15.8	ServiceParams Class Reference	253
15.8.1	Detailed Description	253
15.8.2	Constructor & Destructor Documentation	253
15.8.2.1	ServiceParams()	253
15.8.2.2	ServiceParams(ServiceParams other)	253
15.8.3	Member Function Documentation	253
15.8.3.1	datareader_qos(DataReaderQos qos)	253
15.8.3.2	datareader_qos()	254
15.8.3.3	datawriter_qos(DataWriterQos qos)	254
15.8.3.4	datawriter_qos()	254
15.8.3.5	domain_participant(DomainParticipant part)	254
15.8.3.6	domain_participant()	254
15.8.3.7	instance_name(String instance_name)	254
15.8.3.8	instance_name()	254
15.8.3.9	publisher(Publisher publisher)	254
15.8.3.10	publisher()	254
15.8.3.11	reply_topic_name(String rep_topic)	254
15.8.3.12	reply_topic_name()	254
15.8.3.13	request_topic_name(String req_topic)	255
15.8.3.14	request_topic_name()	255
15.8.3.15	service_name(String service_name)	255
15.8.3.16	service_name()	255
15.8.3.17	subscriber(Subscriber subscriber)	255
15.8.3.18	subscriber()	255
15.9	ServiceProxy< TReq extends RequestType, TRep extends ReplyType > Class Template Reference	256
15.9.1	Detailed Description	256
15.9.2	Member Function Documentation	256
15.9.2.1	bind(String i_name)	256
15.9.2.2	close()	257
15.9.2.3	get_bound_instance_name()	257
15.9.2.4	get_discovered_service_instances()	257
15.9.2.5	is_bound()	257
15.9.2.6	is_null()	257

15.9.2.7	<code>unbind()</code>	257
15.9.2.8	<code>wait_for_service()</code>	257
15.9.2.9	<code>wait_for_service(Duration_t maxWait)</code>	257
15.9.2.10	<code>wait_for_service(String instanceName)</code>	258
15.9.2.11	<code>wait_for_service(Duration_t max_wait, String instanceName)</code>	258
15.9.2.12	<code>wait_for_services(int count)</code>	258
15.9.2.13	<code>wait_for_services(Duration_t max_wait, int count)</code>	258
15.9.2.14	<code>wait_for_services(Vector< String > instanceNames)</code>	258
15.9.2.15	<code>wait_for_services(Duration_t max_wait, Vector< String > instanceNames)</code>	259
15.10	ListenerBase Interface Reference	260
15.10.1	Detailed Description	260
15.11	ReplierBase Interface Reference	261
15.11.1	Detailed Description	261
15.12	ReplierListener< TReq extends RequestType, TRep extends ReplyType > Interface Template Reference	262
15.12.1	Detailed Description	262
15.12.2	Member Function Documentation	262
15.12.2.1	<code>on_request_available(Replier< TReq, TRep > replier)</code>	262
15.13	RequesterListener< TReq extends RequestType, TRep extends ReplyType > Interface Template Reference	263
15.13.1	Detailed Description	263
15.13.2	Member Function Documentation	263
15.13.2.1	<code>on_reply_available(Requester< TReq, TRep > requester)</code>	263
15.14	RPCEntity Interface Reference	264
15.14.1	Detailed Description	264
15.14.2	Member Function Documentation	264
15.14.2.1	<code>close()</code>	264
15.14.2.2	<code>is_null()</code>	264
15.15	SimpleReplierListener< TReq extends RequestType, TRep extends ReplyType > Interface Template Reference	265
15.15.1	Detailed Description	265
15.15.2	Member Function Documentation	265
15.15.2.1	<code>process_request(Sample< TReq > sample, SampleIdentity_t sample_id)</code>	265
15.16	SimpleRequesterListener< TRep extends ReplyType > Interface Template Reference	266
15.16.1	Detailed Description	266
15.16.2	Member Function Documentation	266
15.16.2.1	<code>process_reply(Sample< TRep > sample, SampleIdentity_t sample_identity)</code>	266
15.17	ServiceStatus Enum Reference	267

15.17.1 Detailed Description	267
15.17.2 Member Data Documentation	267
15.17.2.1 CLOSED	267
15.17.2.2 PAUSED	267
15.17.2.3 RUNNING	267
16 CoreDX DDS DynamicTypes	269
17 Data Structure Index	271
17.1 Class List	271
18 Not Yet Supported	279
19 Deprecated List	281
20 Hierarchical Index	283
20.1 Class Hierarchy	283
21 Namespace Details	287
21.1 Packages	287
21.2 Package com.toc.coredx.DDS.rpc	287
21.2.1 Detailed Description	287
Index	288

Chapter 1

Introduction

Welcome to the CoreDX DDS for Java API documentation from Twin Oaks Computing, Inc.

Introduction

CoreDX DDS is a small-footprint, high-performance communications middleware compliant with the OMG Data Distribution Service (DDS) standard. CoreDX DDS supports multiple hardware architectures and operating systems, and is intended to facilitate the development of robust, near real-time, highly distributed systems.

This is the **CoreDX DDS for Java Reference Manual**. It provides a detailed reference for the CoreDX Data Distribution Service implementation from Twin Oaks Computing, Inc. The manual includes documentation on all of the CoreDX DDS data types and Application Programming Interface (API) routines.

The **CoreDX DDS Programmers Guide** provides more information on using the CoreDX DDS API and related tools to produce a complete DDS enabled application.

The CoreDX DDS software provides a high-throughput, standards compliant, data communications infrastructure. CoreDX DDS offers the tools you need to realize Open Architecture goals. Built with a focus on performance, the CoreDX DDS software delivers a quality implementation of the OMG Data Distribution Service (DDS) standard.

The CoreDX DDS software implements the essential Data-Centric Publish-Subscribe (DCPS) communications layer as documented in the OMG DDS Standard. This standalone package, provides everything needed to integrate QoS enabled, Publish-Subscribe messaging into an application. The core software is written in the C language, and is optimized to be small and fast. The core package includes C and C++ language bindings for application integration. This reference manual describes the CoreDX DDS "Java" language binding.

Contents

The reference documentation includes information on the following API constructs:

1. Entities. This includes the primary objects with which an application must interact to enable DDS publish-subscribe communications.
 - [DomainParticipantFactory](#)
 - [DomainParticipant](#)
 - [Topic](#)
 - [ContentFilteredTopic](#)

- [MultiTopic](#)
- [Publisher](#)
- [Subscriber](#)
- [DataReader](#)
- [DataWriter](#)

2. Quality of Service. This section documents the Quality of Service (QoS) structures that configure the behavior of the CoreDX DDS middleware.

- [DomainParticipantFactoryQos](#)
- [DomainParticipantQos](#)
- [TopicQos](#)
- [PublisherQos](#)
- [SubscriberQos](#)
- [DataReaderQos](#)
- [DataWriterQos](#)

3. Listeners and Events. This section covers the various structures and concepts involved in delivering events to the application.

- [DomainParticipantListener](#)
- [TopicListener](#)
- [PublisherListener](#)
- [SubscriberListener](#)
- [DataReaderListener](#)
- [DataWriterListener](#)

4. Status. This section describes the types of status maintained by the infrastructure.

- [InconsistentTopicStatus](#)
- [LivelinessChangedStatus](#)
- [LivelinessLostStatus](#)
- [OfferedDeadlineMissedStatus](#)
- [OfferedIncompatibleQosStatus](#)
- [PublicationMatchedStatus](#)
- [RequestedDeadlineMissedStatus](#)
- [RequestedIncompatibleQosStatus](#)
- [SampleLostStatus](#)
- [SampleRejectedStatus](#)
- [SubscriptionMatchedStatus](#)

5. RPC over DDS

- [ClientEndpoint](#)
 - [ClientParams](#)
 - [ListenerBase](#)
 - [Replier](#)
 - [ReplierBase](#)
-

- [ReplierListener](#)
- [ReplierParams](#)
- [Requester](#)
- [RequesterListener](#)
- [RequesterParams](#)
- [RPCEntity](#)
- [ServiceEndpoint](#)
- [ServiceParams](#)
- [ServiceProxy](#)
- [SimpleReplierListener](#)
- [SimpleRequesterListener](#)

6. [Transports](#). This section includes [Transport](#) objects.

- [UDP Transport](#)
- [TCP Transport](#)
- [LMT Transport \(local machine\)](#)

7. [Miscellaneous](#). This section includes miscellaneous support objects.

- [Condition](#)
- [GuardCondition](#)
- [StatusCondition](#)
- [QueryCondition](#)
- [WaitSet](#)

Intended Audience

This document is intended for software developers who are integrating the CoreDX DDS software into their application(s). The reference manual assumes that the reader is competent in programming languages and software development concepts. CoreDX DDS supports multiple languages, and this reference manual focuses on the Java programming language.

1.1 DDS Entities

Classes

- class [ContentFilteredTopic](#)
[ContentFilteredTopic](#) provides a topic that may include data filtered from a related [Topic](#).
- class [DataReader](#)
The [DataReader](#) entity allows the application to subscribe to and read data.
- class [DataWriter](#)
The [DataWriter](#) entity provides an interface for the application to publish (write) data. The [DataWriter](#) is an abstract class that is extended to support a particular data type required by the application. A [DataReader](#) is associated with, and writes on, a single [Topic](#).
- class [DomainEntity](#)
Base class for all DDS Domain Entities.
- class [DomainParticipant](#)
The [DomainParticipant](#) is used to configure, create and destroy [Publisher](#), [Subscriber](#) and [Topic](#) objects.
- class [DomainParticipantFactory](#)
[DomainParticipantFactory](#) constructs [DomainParticipants](#). The.
- class [Entity](#)
Base class for all DDS Entities.
- class [FooDataReader](#)
The [FooDataReader](#) is an example specialized [DataReader](#) (generated by the `coredx_ddl` compiler) for reading data samples of type 'Foo'.
- class [FooDataWriter](#)
The [FooDataWriter](#) is an example specialized [DataWriter](#) (generated by the `coredx_ddl` compiler) for writing data samples of type 'Foo'.
- class [FooTypeSupport](#)
The [FooTypeSupport](#) class implements the [TypeSupport](#) interface for the Data Type 'Foo'.
- class [MultiTopic](#)
[MultiTopic](#) provides a topic that may include data from multiple [Topics](#).
- class [Publisher](#)
The [Publisher](#) configures, creates, manages and destroys [DataWriters](#).
- class [Subscriber](#)
The [Subscriber](#) configures, creates, manages and destroys [DataReaders](#).
- class [Topic](#)
*[Topic](#) is the basic description of data to be published or subscribed. A topic is identified by a **name** and a **type**. A [Topic](#) is created by calling [DomainParticipant.create_topic\(\)](#). Prior to creating a [Topic](#), the associated data type must be registered with the [DomainParticipant](#) via a call to the [TypeSupportXYZ.register_type\(\)](#) function. [The `register_type()` function is auto-generated 'type-specific' code.].*

1.1.1 Detailed Description

1.2 DDS Quality of Service

Classes

- class [DataReaderQos](#)
Structure that holds [DataReader](#) Quality of Service policies.
- class [DataWriterQos](#)
Structure that holds [DataWriter](#) Quality of Service policies.
- class [DomainParticipantFactoryQos](#)
Structure that holds [DomainParticipantFactory](#) Quality of Service policies.
- class [DomainParticipantQos](#)
Structure that holds [DomainParticipant](#) Quality of Service policies.
- class [PublisherQos](#)
Structure that holds [Publisher](#) Quality of Service policies.
- class [QosProvider](#)
[QosProvider](#) loads QoS settings from a library and provides interfaces to access entity specific QoS Policies.
- class [SubscriberQos](#)
Structure that holds [DDS_Subscriber](#) Quality of Service policies.
- class [TopicQos](#)
Structure that holds [DDS_Topic](#) Quality of Service policies.

Functions

- static String [qos_policy_str](#) (int qos_policy_id)
 - [QosProvider](#) (String uri, String profile)
Constructor. Creates a new [QosProvider](#) instance that loads QoS settings from a library.
 - void [cleanup](#) ()
"Destructor". Returns a [QosProvider](#) back to the factory.
 - DomainParticipantFactoryQos [get_participantfactory_qos](#) (String id)
Access a [DomainParticipantFactory](#) QoS.
 - DomainParticipantQos [get_participant_qos](#) (String id)
Access a [DomainParticipant](#) QoS.
 - TopicQos [get_topic_qos](#) (String id)
Access a [Topic](#) QoS.
 - SubscriberQos [get_subscriber_qos](#) (String id)
Access a [Subscriber](#) QoS.
 - PublisherQos [get_publisher_qos](#) (String id)
Access a [Publisher](#) QoS.
 - DataReaderQos [get_datareader_qos](#) (String id)
Access a [DataReader](#) QoS.
 - DataWriterQos [get_datawriter_qos](#) (String id)
Access a [DataWriter](#) QoS.
-

Variables

- static final int [USERDATA_QOS_POLICY_ID](#) = 1
- static final int [DURABILITY_QOS_POLICY_ID](#) = 2
- static final int [PRESENTATION_QOS_POLICY_ID](#) = 3
- static final int [DEADLINE_QOS_POLICY_ID](#) = 4
- static final int [LATENCYBUDGET_QOS_POLICY_ID](#) = 5
- static final int [OWNERSHIP_QOS_POLICY_ID](#) = 6
- static final int [OWNERSHIPSTRENGTH_QOS_POLICY_ID](#) = 7
- static final int [LIVELINESS_QOS_POLICY_ID](#) = 8
- static final int [TIMEBASEDFILTER_QOS_POLICY_ID](#) = 9
- static final int [PARTITION_QOS_POLICY_ID](#) = 10
- static final int [RELIABILITY_QOS_POLICY_ID](#) = 11
- static final int [DESTINATIONORDER_QOS_POLICY_ID](#) = 12
- static final int [HISTORY_QOS_POLICY_ID](#) = 13
- static final int [RESOURCELIMITS_QOS_POLICY_ID](#) = 14
- static final int [ENTITYFACTORY_QOS_POLICY_ID](#) = 15
- static final int [WRITERDATAIFECYCLE_QOS_POLICY_ID](#) = 16
- static final int [READERDATAIFECYCLE_QOS_POLICY_ID](#) = 17
- static final int [TOPICDATA_QOS_POLICY_ID](#) = 18
- static final int [GROUPDATA_QOS_POLICY_ID](#) = 19
- static final int [TRANSPORTPRIORITY_QOS_POLICY_ID](#) = 20
- static final int [LIFESPAN_QOS_POLICY_ID](#) = 21
- static final int [DURABILITYSERVICE_QOS_POLICY_ID](#) = 22
- static final int [DATA_REPRESENTATION_QOS_POLICY_ID](#) = 23
- static final DataReaderQos [DATAREADER_QOS_DEFAULT](#) = null
- static final DataWriterQos [DATAWRITER_QOS_DEFAULT](#) = null
- static final DomainParticipantQos [PARTICIPANT_QOS_DEFAULT](#) = null
- static final PublisherQos [PUBLISHER_QOS_DEFAULT](#) = null
- static final SubscriberQos [SUBSCRIBER_QOS_DEFAULT](#) = null
- static final TopicQos [TOPIC_QOS_DEFAULT](#) = null

1.2.1 Detailed Description

1.2.2 Function Documentation

1.2.2.1 void cleanup ()

"Destructor". Returns a [QosProvider](#) back to the factory.

This will reclaim any resources allocated by the [QosProvider](#).

1.2.2.2 DataReaderQos get_datareader_qos (String id)

Access a [DataReader](#) QoS.

If 'id' is NULL, then the first instance of [DataReaderQos](#) in the library is returned. If 'id' is not NULL, then the 'name' field of the QoS policy is checked, and the first match is returned.

1.2.2.3 `DataReaderQos get_datawriter_qos ( String id )`

Access a [DataReader](#) QoS.

If 'id' is NULL, then the first instance of [DataReaderQos](#) in the library is returned. If 'id' is not NULL, then the 'name' field of the QoS policy is checked, and the first match is returned.

1.2.2.4 `DomainParticipantQos get_participant_qos ( String id )`

Access a [DomainParticipant](#) QoS.

If 'id' is NULL, then the first instance of [DomainParticipantQos](#) in the library is returned. If 'id' is not NULL, then the 'name' field of the QoS policy is checked, and the first match is returned.

1.2.2.5 `DomainParticipantFactoryQos get_participantfactory_qos ( String id )`

Access a [DomainParticipantFactory](#) QoS.

If 'id' is NULL, then the first instance of [DomainParticipantFactoryQos](#) in the library is returned. If 'id' is not NULL, then the 'name' field of the [DomainParticipantFactoryQos](#) is checked, and the first match is returned.

1.2.2.6 `PublisherQos get_publisher_qos ( String id )`

Access a [Publisher](#) QoS.

If 'id' is NULL, then the first instance of [PublisherQos](#) in the library is returned. If 'id' is not NULL, then the 'name' field of the QoS policy is checked, and the first match is returned.

1.2.2.7 `SubscriberQos get_subscriber_qos ( String id )`

Access a [Subscriber](#) QoS.

If 'id' is NULL, then the first instance of [SubscriberQos](#) in the library is returned. If 'id' is not NULL, then the 'name' field of the QoS policy is checked, and the first match is returned.

1.2.2.8 `TopicQos get_topic_qos ( String id )`

Access a [Topic](#) QoS.

If 'id' is NULL, then the first instance of [TopicQos](#) in the library is returned. If 'id' is not NULL, then the 'name' field of the QoS policy is checked, and the first match is returned.

1.2.2.9 `static String qos_policy_str ( int qos_policy_id ) [static]`

Returns a string (qos_policy_strings) describing the provided QoS Policy ID.

1.2.2.10 `QosProvider ( String uri, String profile )`

Constructor. Creates a new [QosProvider](#) instance that loads QoS settings from a library.

The QoS library source is indicated by 'uri'. Currently CoreDX DDS supports only the 'file:/// ' type URI. The [QosProvider](#) selects QoS policies from the library based on the 'profile' parameter.

1.2.3 Variable Documentation

- 1.2.3.1 `final int DATA_REPRESENTATION_QOS_POLICY_ID = 23` `[static]`
 - 1.2.3.2 `final DataReaderQos DATAREADER_QOS_DEFAULT = null` `[static]`
 - 1.2.3.3 `final DataWriterQos DATAWRITER_QOS_DEFAULT = null` `[static]`
 - 1.2.3.4 `final int DEADLINE_QOS_POLICY_ID = 4` `[static]`
 - 1.2.3.5 `final int DESTINATIONORDER_QOS_POLICY_ID = 12` `[static]`
 - 1.2.3.6 `final int DURABILITY_QOS_POLICY_ID = 2` `[static]`
 - 1.2.3.7 `final int DURABILITYSERVICE_QOS_POLICY_ID = 22` `[static]`
 - 1.2.3.8 `final int ENTITYFACTORY_QOS_POLICY_ID = 15` `[static]`
 - 1.2.3.9 `final int GROUPDATA_QOS_POLICY_ID = 19` `[static]`
 - 1.2.3.10 `final int HISTORY_QOS_POLICY_ID = 13` `[static]`
 - 1.2.3.11 `final int LATENCYBUDGET_QOS_POLICY_ID = 5` `[static]`
 - 1.2.3.12 `final int LIFESPAN_QOS_POLICY_ID = 21` `[static]`
 - 1.2.3.13 `final int LIVELINESS_QOS_POLICY_ID = 8` `[static]`
 - 1.2.3.14 `final int OWNERSHIP_QOS_POLICY_ID = 6` `[static]`
 - 1.2.3.15 `final int OWNERSHIPSTRENGTH_QOS_POLICY_ID = 7` `[static]`
 - 1.2.3.16 `final DomainParticipantQos PARTICIPANT_QOS_DEFAULT = null` `[static]`
 - 1.2.3.17 `final int PARTITION_QOS_POLICY_ID = 10` `[static]`
 - 1.2.3.18 `final int PRESENTATION_QOS_POLICY_ID = 3` `[static]`
 - 1.2.3.19 `final PublisherQos PUBLISHER_QOS_DEFAULT = null` `[static]`
 - 1.2.3.20 `final int READERDATA_LIFECYCLE_QOS_POLICY_ID = 17` `[static]`
 - 1.2.3.21 `final int RELIABILITY_QOS_POLICY_ID = 11` `[static]`
 - 1.2.3.22 `final int RESOURCELIMITS_QOS_POLICY_ID = 14` `[static]`
 - 1.2.3.23 `final SubscriberQos SUBSCRIBER_QOS_DEFAULT = null` `[static]`
 - 1.2.3.24 `final int TIMEBASEDFILTER_QOS_POLICY_ID = 9` `[static]`
 - 1.2.3.25 `final TopicQos TOPIC_QOS_DEFAULT = null` `[static]`
-

1.2.3.26 `final int TOPICDATA_QOS_POLICY_ID = 18` `[static]`

1.2.3.27 `final int TRANSPORTPRIORITY_QOS_POLICY_ID = 20` `[static]`

1.2.3.28 `final int USERDATA_QOS_POLICY_ID = 1` `[static]`

1.2.3.29 `final int WRITERDATALIFECYCLE_QOS_POLICY_ID = 16` `[static]`

1.3 DDS Conditions, Listeners, and WaitSets

Modules

- [DDS Listeners](#)
- [DDS Conditions](#)
- [DDS WaitSets](#)

Variables

- static final int [READ_SAMPLE_STATE](#) = (0x0001)
- static final int [NOT_READ_SAMPLE_STATE](#) = (0x0002)
- static final int [ANY_SAMPLE_STATE](#) = (0x00FF)
- static final int [NEW_VIEW_STATE](#) = (0x0001)
- static final int [NOT_NEW_VIEW_STATE](#) = (0x0002)
- static final int [ANY_VIEW_STATE](#) = (0x00FF)
- static final int [ALIVE_INSTANCE_STATE](#) = (0x0001)
- static final int [NOT_ALIVE_DISPOSED_INSTANCE_STATE](#) = (0x0002)
- static final int [NOT_ALIVE_NO_WRITERS_INSTANCE_STATE](#) = (0x0004)
- static final int [NOT_ALIVE_INSTANCE_STATE](#) = (0x0006)
- static final int [ANY_INSTANCE_STATE](#) = (0x00FF)

1.3.1 Detailed Description

1.3.2 Variable Documentation

1.3.2.1 final int [ALIVE_INSTANCE_STATE](#) = (0x0001) [static]

1.3.2.2 final int [ANY_INSTANCE_STATE](#) = (0x00FF) [static]

1.3.2.3 final int [ANY_SAMPLE_STATE](#) = (0x00FF) [static]

1.3.2.4 final int [ANY_VIEW_STATE](#) = (0x00FF) [static]

1.3.2.5 final int [NEW_VIEW_STATE](#) = (0x0001) [static]

1.3.2.6 final int [NOT_ALIVE_DISPOSED_INSTANCE_STATE](#) = (0x0002) [static]

1.3.2.7 final int [NOT_ALIVE_INSTANCE_STATE](#) = (0x0006) [static]

1.3.2.8 final int [NOT_ALIVE_NO_WRITERS_INSTANCE_STATE](#) = (0x0004) [static]

1.3.2.9 final int [NOT_NEW_VIEW_STATE](#) = (0x0002) [static]

1.3.2.10 final int [NOT_READ_SAMPLE_STATE](#) = (0x0002) [static]

1.3.2.11 final int [READ_SAMPLE_STATE](#) = (0x0001) [static]

1.4 DDS Listeners

Classes

- interface [DataReaderListener](#)

The [DataReaderListener](#) provides asynchronous notification of [DataReader](#) events. This listener can be installed during [DataReader](#) creation, `DomainParticipant_create_yyy()`, as well as by calling `DataReader_set_listener()`.

- interface [DataWriterListener](#)

The [DataWriterListener](#) provides asynchronous notification of [DataWriter](#) events. This listener can be installed during [DataWriter](#) creation, `Publisher_create_datawriter()`, as well as by calling `DataWriter_set_listener()`.

- interface [DomainParticipantListener](#)

- interface [PublisherListener](#)

- interface [SubscriberListener](#)

The [SubscriberListener](#) provides asynchronous notification of [Subscriber](#) events. This listener can be installed during [Subscriber](#) creation, `DomainParticipant_create_subscriber()` as well as by calling `Subscriber_set_listener()`.

- interface [TopicListener](#)

The `DDS_TopicListener` provides asynchronous notification of `DDS_Topic` events. This listener can be installed during [Topic](#) creation (`DDS_DomainParticipant_create_topic()` and related) as well as by calling `DDS_Topic_set_listener()`.

1.4.1 Detailed Description

1.5 DDS Conditions

Classes

- class [Condition](#)

A [Condition](#) can be added to a `DDS_WaitSet` to provide synchronous event notification.

- class [GuardCondition](#)

*A [GuardCondition](#) is a [Condition](#) where the **trigger_value** is under application control.*

- class [QueryCondition](#)

*The **trigger_value** is driven by the data available, after applying the filter, in the associated [DataReader](#).*

- class [ReadCondition](#)

A [ReadCondition](#) is a specialized [Condition](#) associated with a [DataReader](#).

1.5.1 Detailed Description

1.6 DDS WaitSets

Classes

- class [WaitSet](#)

*A `DDS_WaitSet` maintains a set of [Condition](#) objects and allows the application to wait until one or more of them have a ***trigger_value*** of `TRUE`.*

1.6.1 Detailed Description

1.7 DDS Status Structures

Classes

- class [InconsistentTopicStatus](#)
[InconsistentTopicStatus](#) provides status related to the `on_inconsistent_topic` listener methods of the [TopicListener](#) structure.
- class [LivelinessChangedStatus](#)
Status related to the `on_liveliness_changed` listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.
- class [LivelinessLostStatus](#)
Status related to the `on_liveliness_lost` listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.
- class [OfferedDeadlineMissedStatus](#)
Status related to the `on_offered_deadline_missed` listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.
- class [OfferedIncompatibleQosStatus](#)
Status related to the `on_offered_incompatible_qos` listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.
- class [PublicationMatchedException](#)
Status related to the `on_publication_matched` listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.
- class [RequestedDeadlineMissedStatus](#)
Status related to the `on_requested_deadline_missed` listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.
- class [SampleLostStatus](#)
Status related to the `on_sample_lost` listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.
- class [SampleRejectedStatus](#)
Status related to the `on_sample_rejected` listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.
- class [SubscriptionMatchedException](#)
Status related to the `on_subscription_matched` listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

Variables

- static final int [INCONSISTENT_TOPIC_STATUS](#) = (0x0001)
- static final int [OFFERED_DEADLINE_MISSED_STATUS](#) = (0x0002)
- static final int [REQUESTED_DEADLINE_MISSED_STATUS](#) = (0x0004)
- static final int [OFFERED_INCOMPATIBLE_QOS_STATUS](#) = (0x0008)
- static final int [REQUESTED_INCOMPATIBLE_QOS_STATUS](#) = (0x0010)
- static final int [SAMPLE_LOST_STATUS](#) = (0x0020)
- static final int [SAMPLE_REJECTED_STATUS](#) = (0x0040)
- static final int [DATA_ON_READERS_STATUS](#) = (0x0080)
- static final int [DATA_AVAILABLE_STATUS](#) = (0x0100)
- static final int [LIVELINESS_LOST_STATUS](#) = (0x0200)
- static final int [LIVELINESS_CHANGED_STATUS](#) = (0x0400)
- static final int [PUBLICATION_MATCHED_STATUS](#) = (0x0800)
- static final int [SUBSCRIPTION_MATCHED_STATUS](#) = (0x1000)
- static final int [NO_STATUS](#) = (0)
- static final int [STATUS_MASK_NONE](#) = (0)
- static final int [ALL_STATUS](#) = (0xFFFF)

1.7.1 Detailed Description

- [InconsistentTopicStatus](#)
- [LivelinessChangedStatus](#)
- [LivelinessLostStatus](#)
- [OfferedDeadlineMissedStatus](#)
- [OfferedIncompatibleQosStatus](#)
- [PublicationMatchedStatus](#)
- [RequestedDeadlineMissedStatus](#)
- [RequestedIncompatibleQosStatus](#)
- [SampleLostStatus](#)
- [SampleRejectedStatus](#)
- [SubscriptionMatchedStatus](#)

1.7.2 Variable Documentation

1.7.2.1 final int ALL_STATUS = (0xFFFF) [static]

1.7.2.2 final int DATA_AVAILABLE_STATUS = (0x0100) [static]

1.7.2.3 final int DATA_ON_READERS_STATUS = (0x0080) [static]

1.7.2.4 final int INCONSISTENT_TOPIC_STATUS = (0x0001) [static]

1.7.2.5 final int LIVELINESS_CHANGED_STATUS = (0x0400) [static]

1.7.2.6 final int LIVELINESS_LOST_STATUS = (0x0200) [static]

1.7.2.7 final int NO_STATUS = (0) [static]

1.7.2.8 final int OFFERED_DEADLINE_MISSED_STATUS = (0x0002) [static]

1.7.2.9 final int OFFERED_INCOMPATIBLE_QOS_STATUS = (0x0008) [static]

1.7.2.10 final int PUBLICATION_MATCHED_STATUS = (0x0800) [static]

1.7.2.11 final int REQUESTED_DEADLINE_MISSED_STATUS = (0x0004) [static]

1.7.2.12 final int REQUESTED_INCOMPATIBLE_QOS_STATUS = (0x0010) [static]

1.7.2.13 final int SAMPLE_LOST_STATUS = (0x0020) [static]

1.7.2.14 final int SAMPLE_REJECTED_STATUS = (0x0040) [static]

1.7.2.15 final int STATUS_MASK_NONE = (0) [static]

1.7.2.16 final int SUBSCRIPTION_MATCHED_STATUS = (0x1000) [static]

1.8 CoreDX DDS Transports

Classes

- class [IpTransportInterface](#)
Defines an interface for use by the [UdpTransport](#).
- class [LmtTransport](#)
An instance of a LMT [Transport](#) that can be added to a [DomainParticipant](#).
- class [LmtTransportConfig](#)
Structure that holds LMT [Transport](#) configuration items.
- class [SslTransport](#)
An instance of a SSL [Transport](#) that can be added to a [DomainParticipant](#).
- class [SslTransportConfig](#)
Structure that holds SSL [Transport](#) configuration items.
- class [TcpTransport](#)
An instance of a TCP [Transport](#) that can be added to a [DomainParticipant](#).
- class [TcpTransportConfig](#)
Structure that holds TCP [Transport](#) configuration items.
- class [Transport](#)
An instance of a [Transport](#) that can be added to a [DomainParticipant](#).
- class [UdpTransport](#)
An instance of a [Transport](#) that can be added to a [DomainParticipant](#).
- class [UdpTransportConfig](#)
Structure that holds UDP [Transport](#) configuration items.

1.8.1 Detailed Description

1.9 CoreDX DDS RPC

Classes

- class [ClientEndpoint](#)< TReq extends RequestType, TRep extends ReplyType >
- class [ClientParams](#)
- interface [ListenerBase](#)
- class [Replier](#)< TReq extends RequestType, TRep extends ReplyType >
- interface [ReplierBase](#)
- interface [ReplierListener](#)< TReq extends RequestType, TRep extends ReplyType >
- class [ReplierParams](#)
- class [Requester](#)< TReq extends RequestType, TRep extends ReplyType >
- interface [RequesterListener](#)< TReq extends RequestType, TRep extends ReplyType >
- class [RequesterParams](#)
- interface [RPCEntity](#)
- class [ServiceEndpoint](#)
- class [ServiceParams](#)
- class [ServiceProxy](#)< TReq extends RequestType, TRep extends ReplyType >
- enum [ServiceStatus](#)
- interface [SimpleReplierListener](#)< TReq extends RequestType, TRep extends ReplyType >
- interface [SimpleRequesterListener](#)< TRep extends ReplyType >

1.9.1 Detailed Description

1.10 CoreDX DDS DynamicType

1.11 DDS Builtin Entities and Data Types

Classes

- class [ParticipantBuiltinTopicData](#)

The [ParticipantBuiltinTopicData](#) is a built-in Data Type that represents Participant Discovery information.

- class [ParticipantBuiltinTopicDataDataReader](#)

The [ParticipantBuiltinTopicDataDataReader](#) is a built-in [DataReader](#) that can be used to access Participant Discovery information. The reader can be accessed through the special 'builtin' [Subscriber](#) via the [DomainParticipant.get_builtin_subscriber\(\)](#) routine.

- class [PublicationBuiltinTopicData](#)

The [PublicationBuiltinTopicData](#) is a built-in Data Type that represents Publication Discovery information.

- class [PublicationBuiltinTopicDataDataReader](#)

The [PublicationBuiltinTopicDataDataReader](#) is a built-in [DataReader](#) that can be used to access Publication Discovery information. The reader can be accessed through the special 'builtin' [Subscriber](#) via the [DomainParticipant.get_builtin_subscriber\(\)](#) routine.

- class [SubscriptionBuiltinTopicData](#)

The [SubscriptionBuiltinTopicData](#) is a built-in Data Type that represents Subscription Discovery information.

- class [SubscriptionBuiltinTopicDataDataReader](#)

The [SubscriptionBuiltinTopicDataDataReader](#) is a built-in [DataReader](#) that can be used to access Subscription Discovery information.

1.11.1 Detailed Description

Chapter 2

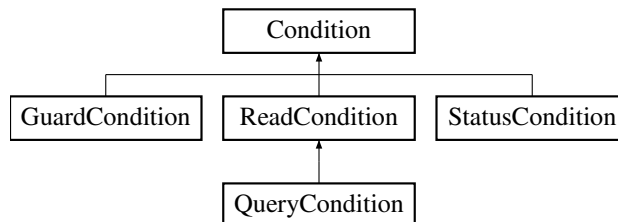
Primary DDS Entities and Types

This section documents the DDS Entities and Types that the application developer will almost always interact with.

2.1 Condition Class Reference

A [Condition](#) can be added to a DDS_WaitSet to provide synchronous event notification.

Inheritance diagram for Condition:



Public Member Functions

- boolean [get_trigger_value](#) ()

2.1.1 Detailed Description

A [Condition](#) can be added to a DDS_WaitSet to provide synchronous event notification.

A [Condition](#) has a **trigger_value** which can be TRUE or FALSE.

2.1.2 Member Function Documentation

2.1.2.1 boolean [get_trigger_value](#) ()

This routine returns the current value of the **trigger_value** in [Condition c](#).

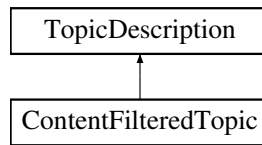
A non-zero return value indicates that the **trigger_value** is TRUE.

A zero return value indicates that the **trigger_value** is FALSE.

2.2 ContentFilteredTopic Class Reference

[ContentFilteredTopic](#) provides a topic that may include data filtered from a related [Topic](#).

Inheritance diagram for ContentFilteredTopic:



Public Member Functions

- [DomainParticipant](#) [get_participant](#) ()
*This operation returns the parent **DomainParticipant** of the [ContentFilteredTopic](#).*
- String [get_type_name](#) ()
*This operation returns **type_name** of the [ContentFilteredTopic](#).*
- String [get_name](#) ()
*This operation returns **topic name** of the [ContentFilteredTopic](#).*
- [Topic](#) [get_related_topic](#) ()
This returns the real [Topic](#) associated with the [ContentFilteredTopic](#).
- Vector [get_expression_parameters](#) ()
This accesses the current set of parameters used by the [ContentFilteredTopic](#).
- [ReturnCode_t](#) [set_expression_parameters](#) (Vector filter_parameters)
This specifies a new set of parameters for use with the filter_expression.

2.2.1 Detailed Description

[ContentFilteredTopic](#) provides a topic that may include data filtered from a related [Topic](#).

2.2.2 Member Function Documentation

2.2.2.1 Vector [get_expression_parameters](#) ()

This accesses the current set of parameters used by the [ContentFilteredTopic](#).

The **parameters** String Sequence is populated with the current set of parameters.

2.2.2.2 [Topic](#) [get_related_topic](#) ()

This returns the real [Topic](#) associated with the [ContentFilteredTopic](#).

That is, the [Topic](#) provided when the [ContentFilteredTopic](#) was created.

2.2.2.3 ReturnCode_t set_expression_parameters (Vector *filter_parameters*)

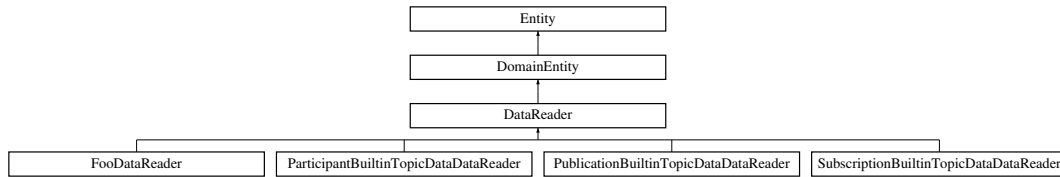
This specifies a new set of parameters for use with the filter_expression.

The **filter_expression** is an SQL like condition expression, and the **parameters** argument provides optional parameters that are referenced by the **filter_expression**. The syntax for referring to parameters in a filter_expression is the percent sign " " followed by a number. The number is the index of the parameter in the **filter_parameters** sequence. Parameters are counted starting at zero. So, "%0" refers to the first parameter, and "%4" refers to the fifth parameter. Using this syntax, the expression "x<%0" would test the value of 'x' against the first parameter in the sequence.

2.3 DataReader Class Reference

The [DataReader](#) entity allows the application to subscribe to and read data.

Inheritance diagram for DataReader:



Public Member Functions

- `ReturnCode_t enable ()`
- `ReadCondition create_readcondition (long sample_states, long view_states, long instance_states)`
- `QueryCondition create_querycondition (long sample_states, long view_states, long instance_states, String query_expression, Vector query_parameters)`
Creates a DDS_QueryCondition.
- `ReturnCode_t delete_readcondition (ReadCondition rc)`
- `ReturnCode_t delete_contained_entities ()`
- `ReturnCode_t set_qos (DataReaderQos qos)`
- `ReturnCode_t get_qos (DataReaderQos qos)`
- `ReturnCode_t set_listener (DataReaderListener new_listener, long mask)`
- `DataReaderListener get_listener ()`
- `TopicDescription get_topicdescription ()`
- `Subscriber get_subscriber ()`
- `ReturnCode_t get_sample_rejected_status (SampleRejectedStatus status)`
- `ReturnCode_t get_liveliness_changed_status (LivelinessChangedStatus status)`
- `ReturnCode_t get_requested_deadline_missed_status (RequestedDeadlineMissedStatus status)`
- `ReturnCode_t get_requested_incompatible_qos_status (RequestedIncompatibleQosStatus status)`
- `ReturnCode_t get_subscription_matched_status (SubscriptionMatchedStatus status)`
- `ReturnCode_t get_sample_lost_status (SampleLostStatus status)`
- `ReturnCode_t wait_for_historical_data (Duration_t max_wait)`
- `ReturnCode_t get_matched_publications (InstanceHandleSeq publication_handles)`
- `ReturnCode_t get_matched_publication_data (PublicationBuiltinTopicData publication_data, InstanceHandle_t publication_handle)`
- `ReturnCode_t get_guid (GUID_t g)`

Access the GUID which uniquely identifies this reader.

2.3.1 Detailed Description

The [DataReader](#) entity allows the application to subscribe to and read data.

The DataReader is an abstract class that is extended to support a particular data type required by the application. A [DataReader](#) is associated with a single [TopicDescription](#) ([Topic](#), [MultiTopic](#), or [ContentFilteredTopic](#)).

See also

[com.toc.coredx.DDS.FooDataReader](#)

2.3.2 Member Function Documentation

2.3.2.1 QueryCondition create_querycondition (long sample_states, long view_states, long instance_states, String query_expression, Vector query_parameters)

Creates a DDS_QueryCondition.

The returned [QueryCondition](#) can be used as an argument to [read_w_condition\(\)](#) or [take_w_condition\(\)](#).

The \b query_expression is an SQL like condition expression, and \b query_parameters provide optional parameters that are referenced by the \b query_expression. The syntax of the query expression is similar to the WHERE clause in SQL. For example "x<4" is a valid query expression. It would test that data member 'x' is less than value '4'. If the query expression evaluates to TRUE, then the data sample will be available, otherwise the data sample will be 'filtered' (excluded).

CoreDX %DDS supports the 'LIKE' operator (and NOT LIKE) for regular expression string matching. The pattern string in a LIKE clause can contain '%' to match zero or more characters, '_' to match a single character, or '[<characters>]' to match a range of characters.

CoreDX %DDS also includes support for the 'IN' operator. This provides a very powerful mechanism for testing that a value appears in a set of values. For example "symbol IN ('ge', 'msft', 'ibm')" will select all samples that have a symbol value of 'ge', 'msft', or 'ibm'. This could also be written as a series of equality tests combined with the OR operator; however, the IN operator is much more efficient. A query that matches on several hundred or even thousands of values can be implemented very efficiently using the 'IN' operator.

The query_expression can refer to parameters. The syntax for parameters is the percent sign '%' followed by a number. The number is the index of the parameter in the \b query_parameters sequence. Parameters are counted starting at zero. So, "%0" refers to the first parameter, and "%4" refers to the fifth parameter. Using this syntax, the expression "x<%0" would test the value of 'x' against the first parameter in the sequence.

See also

[FooDataReader::read_w_condition\(\)](#).

[FooDataReader::take_w_condition\(\)](#).

DDS for a list of available STATE constants which may be used

Not Yet Supported QueryConditions are not yet supported as triggers for a [WaitSet](#).

2.3.2.2 ReadCondition create_readcondition (long sample_states, long view_states, long instance_states)

Creates a [ReadCondition](#) that is associated with this [DataReader](#). The returned condition can be added to a [WaitSet](#) or used in a call to the specialized **read()** or **take()** operations. For example see [DataReaderFoo.read_w_condition\(\)](#);

See also

DDS for a list of available STATE constants which may be used

2.3.2.3 ReturnCode_t delete_contained_entities ()

This operation deletes all the [ReadCondition](#) and [QueryCondition](#) objects previously created by means of the [DataReader.create_readcondition\(\)](#) and [DataReader.create_querycondition\(\)](#) operations.

After successful execution, the application may delete the [Publisher](#) by calling [Subscriber.delete_datareader\(\)](#).

If any of the objects cannot be deleted, this routine will return [ReturnCode_t.RETCODE_PRECONDITION_NOT_MET](#).

2.3.2.4 ReturnCode_t delete_readcondition (ReadCondition rc)

Destroys a [ReadCondition](#) (or [QueryCondition](#)). The provided **a_condition** must have been previously created via a call to [DataReader.create_readcondition\(\)](#) or [DataReader.create_querycondition\(\)](#).

The **a_condition** argument must belong to [DataReader dr](#). Otherwise, the error ReturnCode_t.RETCODE_PRECONDITION_NOT_MET will be returned.

If the [DataReader](#) is actively processing the [ReadCondition](#), this routine will return ReturnCode_t.RETCODE_ERROR; in this case, the [delete_readcondition\(\)](#) call should be re-tried.

2.3.2.5 ReturnCode_t enable ()

Enables the [DataReader](#). A [DataReader](#) is created either enabled or not based on the [SubscriberQos](#) setting **entity_factory**. When a [DataReader](#) is not enabled, only the following sub-set of all [DataReader](#) operations are legal:

- operations to get and set QoS policies,
- [get_statuscondition\(\)](#),
- [get_status_changes\(\)](#),

Any other operation may return the ReturnCode_t.RETCODE_NOT_ENABLED error. [DataReader_enable\(\)](#) may be called on an already enabled [DataReader](#) [it will have no effect].

2.3.2.6 DataReaderListener get_listener ()

This operation returns the currently installed [DataReaderListener](#).

2.3.2.7 ReturnCode_t get_liveliness_changed_status (LivelinessChangedStatus status)

Provides access to the current [LivelinessChangedStatus](#) of the [DataReader](#). As a side-effect, this routine will reset the **total_count_change** status field to zero.

2.3.2.8 ReturnCode_t get_matched_publication_data (PublicationBuiltinTopicData publication_data, InstanceHandle_t publication_handle)

This operation returns data that describes a particular matched [DataWriter](#) identified by **publication_handle**. An appropriate handle can be obtained through a call to [DataReader.get_matched_publications\(\)](#).

If **publication_handle** does not identify a currently matched [DataWriter](#), this routine will return ReturnCode_t.RETCODE_PRECONDITION_NOT_MET.

2.3.2.9 ReturnCode_t get_matched_publications (InstanceHandleSeq publication_handles)

This operation retrieves the list of DataWriters currently matched with this [DataReader dr](#). This list will include the handles that identify DataWriters which have matching [Topic](#) and compatible QoS with [DataReader](#).

If a [DataWriter](#) has been ignored by a call to [DomainParticipant.ignore_publication\(\)](#), then it will not appear in the list.

Parameters

<i>publication_handles</i>	A vector that will be populated with InstanceHandle_t(s).
----------------------------	---

2.3.2.10 `ReturnCode_t get_qos ( DataReaderQos qos )`

Returns the current [DataReaderQos](#) settings held in the [DataReader](#) **dr**. This routine copies data from the [DataReader](#) QoS properties into **qos**.

2.3.2.11 `ReturnCode_t get_requested_deadline_missed_status ( RequestedDeadlineMissedStatus status )`

Provides access to the current [RequestedDeadlineMissedStatus](#) of the [DataReader](#). As a side-effect, this routine will reset the **total_count_change** status field to zero.

2.3.2.12 `ReturnCode_t get_requested_incompatible_qos_status ( RequestedIncompatibleQosStatus status )`

Provides access to the current [RequestedIncompatibleQosStatus](#) of the [DataReader](#). As a side-effect, this routine will reset the **total_count_change** status field to zero.

2.3.2.13 `ReturnCode_t get_sample_lost_status ( SampleLostStatus status )`

Provides access to the current [SampleLostStatus](#) of the [DataReader](#). As a side-effect, this routine will reset the **total_count_change** status field to zero.

2.3.2.14 `ReturnCode_t get_sample_rejected_status ( SampleRejectedStatus status )`

Provides access to the current [SampleRejectedStatus](#) of the [DataReader](#). As a side-effect, this routine will reset the **total_count_change** status field to zero.

2.3.2.15 `Subscriber get_subscriber ( )`

Returns the Subscriber that contains [DataReader](#) **dr**.

2.3.2.16 `ReturnCode_t get_subscription_matched_status ( SubscriptionMatchedStatus status )`

Provides access to the current [SubscriptionMatchedStatus](#) of the [DataReader](#). As a side-effect, this routine will reset the **total_count_change** status field to zero.

2.3.2.17 `TopicDescription get_topicdescription ( )`

Returns the [TopicDescription](#) associated with [DataReader](#) **dr**.

2.3.2.18 `ReturnCode_t set_listener ( DataReaderListener new_listener, long mask )`

Installs a [DataReaderListener](#) on [DataReader](#) **dr**. Only one listener may be attached to a [DataReader](#) at a time. A call to **set_listener()** will replace any current listener with **a_listener**.

a_listener can be NULL, which indicates a listener that does nothing.

See also

DDS for a list of available STATUS constants which may be or'ed together to construct the mask.

2.3.2.19 `ReturnCode_t set_qos ( DataReaderQos qos )`

Sets the [DataReaderQos](#) values. These QoS values affect the behavior of the [DataReader](#). This routine may fail if the provided **qos** argument is not internally consistent. In this case, `ReturnCode_t.RETCODE_INCONSISTENT_POLICY` will be returned, and no changes will be made to the [DataReader](#) QoS.

2.3.2.20 `ReturnCode_t wait_for_historical_data ( Duration_t max_wait )`

This routine blocks until all 'historical' data is received.

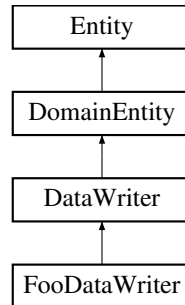
Parameters

<i>max_wait</i>	The maximum amount of time to block while waiting. Can be set to { INFINITE_SEC, INFINITE_NSEC }
-----------------	--

2.4 DataWriter Class Reference

The [DataWriter](#) entity provides an interface for the application to publish (write) data. The [DataWriter](#) is an abstract class that is extended to support a particular data type required by the application. A [DataReader](#) is associated with, and writes on, a single [Topic](#).

Inheritance diagram for DataWriter:



Public Member Functions

- `ReturnCode_t enable ()`
- `ReturnCode_t set_qos (DataWriterQos qos)`
- `ReturnCode_t get_qos (DataWriterQos qos)`
- `ReturnCode_t set_listener (DataWriterListener new_listener, long mask)`
- `DataWriterListener get_listener ()`
- `Topic get_topic ()`
- `Publisher get_publisher ()`
- `ReturnCode_t wait_for_acknowledgments (Duration_t max_wait)`
Block until this writer has received acknowledgements for all written data.
- `ReturnCode_t assert_liveliness ()`
- `ReturnCode_t get_liveliness_lost_status (LivelinessLostStatus status)`
- `ReturnCode_t get_offered_deadline_missed_status (OfferedDeadlineMissedStatus status)`
- `ReturnCode_t get_offered_incompatible_qos_status (OfferedIncompatibleQosStatus status)`
- `ReturnCode_t get_publication_matched_status (PublicationMatchedStatus status)`
- `ReturnCode_t get_matched_subscriptions (InstanceHandleSeq subscription_handles)`
- `ReturnCode_t get_matched_subscription_data (SubscriptionBuiltinTopicData subscription_data, InstanceHandle_t subscription_handle)`
- `ReturnCode_t get_guid (GUID_t g)`

Access the GUID which uniquely identifies this writer.

2.4.1 Detailed Description

The [DataWriter](#) entity provides an interface for the application to publish (write) data. The [DataWriter](#) is an abstract class that is extended to support a particular data type required by the application. A [DataReader](#) is associated with, and writes on, a single [Topic](#).

2.4.2 Member Function Documentation

2.4.2.1 `ReturnCode_t assert_liveliness ( )`

This operation manually asserts the liveliness of the [DataWriter dw](#). This operation is useful if the LIVELINESS QoS setting is `MANUAL_BY_PARTICIPANT_LIVELINESS_QOS` or `MANUAL_BY_TOPIC_LIVELINESS_QOS`; otherwise, it has no effect.

The **write** operation automatically asserts liveliness on the [DataWriter](#) and its [DomainParticipant](#). Therefore, **assert_liveliness** is required only if the application is not writing data frequently enough to satisfy the LIVELINESS setting.

2.4.2.2 `ReturnCode_t enable ( )`

Enables the [DataWriter](#). A [DataWriter](#) is created either enabled or not based on the [PublisherQos](#) setting `entity_factory`. When a [DataWriter](#) is not enabled, only the following sub-set of all [DataWriter](#) operations are legal:

- operations to get and set QoS policies,
- [get_statuscondition\(\)](#),
- [get_status_changes\(\)](#),

Any other operation may return the `RETCODE_NOT_ENABLED` error. `DataWriter_enable()` may be called on an already enabled [DataWriter](#) [it will have no effect].

2.4.2.3 `DataWriterListener get_listener ( )`

This operation returns the currently installed [DataWriterListener](#).

2.4.2.4 `ReturnCode_t get_liveliness_lost_status ( LivelinessLostStatus status )`

Provides access to the current [LivelinessLostStatus](#) of the [DataWriter](#). As a side-effect, this routine will reset the `total_count_change` status field to zero.

2.4.2.5 `ReturnCode_t get_matched_subscription_data ( SubscriptionBuiltinTopicData subscription_data, InstanceHandle_t subscription_handle )`

This operation returns data that describes a particular matched [DataReader](#) identified by `subscription_handle`. An appropriate handle can be obtained through a call to [DataWriter.get_matched_subscriptions\(\)](#).

If `subscription_handle` does not identify a matched [DataReader](#), this routine will return `ReturnCode_t.RETCODE_PRECONDITION_NOT`

2.4.2.6 `ReturnCode_t get_matched_subscriptions ( InstanceHandleSeq subscription_handles )`

This operation retrieves the list of [DataReaders](#) currently matched with the [DataWriter dw](#). This list will include the handles that identify [DataReaders](#) which have matching [Topic](#) and compatible QoS with [DataWriter](#).

If a [DataReader](#) has been ignored by a call to [DomainParticipant.ignore_subscription\(\)](#), then it will not appear in the list.

Parameters

<i>subscription_handles</i>	A vector that will be populated with InstanceHandle_t(s).
-----------------------------	---

2.4.2.7 ReturnCode_t get_offered_deadline_missed_status (OfferedDeadlineMissedStatus status)

Provides access to the current [OfferedDeadlineMissedStatus](#) of the [DataWriter](#). As a side-effect, this routine will reset the **total_count_change** status field to zero.

2.4.2.8 ReturnCode_t get_offered_incompatible_qos_status (OfferedIncompatibleQosStatus status)

Provides access to the current [OfferedIncompatibleQosStatus](#) of the [DataWriter](#). As a side-effect, this routine will reset the **total_count_change** status field to zero.

2.4.2.9 ReturnCode_t get_publication_matched_status (PublicationMatchedStatus status)

Provides access to the current [PublicationMatchedStatus](#) of the [DataWriter](#). As a side-effect, this routine will reset the **total_count_change** and **current_count_change** status fields to zero.

2.4.2.10 Publisher get_publisher ()

Returns the [Publisher](#) that contains [DataWriter dw](#).

2.4.2.11 ReturnCode_t get_qos (DataWriterQos qos)

Returns the current [DataWriterQos](#) settings held in the [DataWriter dw](#). This routines copies data from the [DataWriter](#) QoS properties into **qos**.

2.4.2.12 Topic get_topic ()

Returns the [Topic](#) associated with [DataWriter dw](#).

2.4.2.13 ReturnCode_t set_listener (DataWriterListener new_listener, long mask)

Installs a [DataWriterListener](#) on [DataWriter dw](#). Only one listener may be attached to a [DataWriter](#) at a time. A call to [set_listener\(\)](#) will replace any current listener with **a_listener**.

a_listener can be NULL, which indicates a listener that does nothing.

See also

DDS for a list of available STATUS constants which may be or'ed together to construct the mask.

2.4.2.14 ReturnCode_t set_qos (DataWriterQos qos)

Sets the [DataWriterQos](#) values. These QoS values affect the behavior of the [DataWriter](#). This routine may fail if the provided **qos** argument is not internally consistent. In this case, ReturnCode_t.RETCODE_INCONSISTENT_POLICY will be returned, and no changes will be made to the [DataWriter](#) QoS.

2.4.2.15 `ReturnCode_t wait_for_acknowledgments ( Duration_t max_wait )`

Block until this writer has received acknowledgements for all written data.

This routine will block until all data written by the writer has been acknowledged, or until the 'max_wait' duration has passed. 'max_wait' can be set to INFINITE, in which case this routine may block indefinitely.

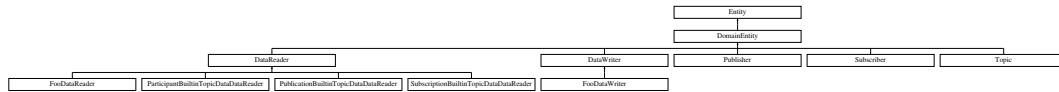
Return values

<i>DDS_RETCODE_TIME_OUT</i>	returned if 'max_wait' passes before all acks are received
<i>DDS_RETCODE_OK</i>	returned if all acks have been received before 'max_wait'

2.5 DomainEntity Class Reference

Base class for all DDS Domain Entities.

Inheritance diagram for DomainEntity:



Additional Inherited Members

2.5.1 Detailed Description

Base class for all DDS Domain Entities.

2.6 DomainParticipantFactory Class Reference

[DomainParticipantFactory](#) constructs DomainParticipants. The.

Public Member Functions

- [DomainParticipant](#) `create_participant` (long *domain_id*, [DomainParticipantQos](#) *qos*, [DomainParticipantListener](#) *listener*, long *mask*)
- `ReturnCode_t` `delete_participant` ([DomainParticipant](#) *participant*)
- [DomainParticipant](#) `lookup_participant` (long *domain_id*)
- `ReturnCode_t` `set_default_participant_qos` ([DomainParticipantQos](#) *qos*)
- `ReturnCode_t` `get_default_participant_qos` ([DomainParticipantQos](#) *qos*)
- `ReturnCode_t` `set_qos` ([DomainParticipantFactoryQos](#) *qos*)
- `ReturnCode_t` `get_qos` ([DomainParticipantFactoryQos](#) *qos*)

Static Public Member Functions

- static [DomainParticipantFactory](#) `get_instance` ()

2.6.1 Detailed Description

[DomainParticipantFactory](#) constructs DomainParticipants. The.

[DomainParticipantFactory](#) is used to configure, create and destroy [DomainParticipant](#) instances.

Author

Twin Oaks Computing, Inc

Since

2.2

2.6.2 Member Function Documentation

2.6.2.1 [DomainParticipant](#) `create_participant` (long *domain_id*, [DomainParticipantQos](#) *qos*, [DomainParticipantListener](#) *listener*, long *mask*)

Create and initialize a [DomainParticipant](#). The caller provides the **domain_id** to which the Participant should belong. The **listener** and **mask** arguments are used to specify a set of callback routines which will be invoked upon detection of certain events.

The **qos** argument specifies the [DomainParticipant](#) Quality of Service settings that should be used when creating the [DomainParticipant](#). It may be specified as **DDS.PARTICIPANT_QOS_DEFAULT** to instruct CoreDX to use the default qos settings held in the [DomainParticipantFactory](#).

This routine will return **NULL** if it fails to create a [DomainParticipant](#).

Parameters

<i>domain_id</i>	The DomainId of the created DomainParticipant .
------------------	---

Parameters

<i>qos</i>	The QoS to use for the create DP.
<i>listener</i>	The listener that will receive event notifications, can be null
<i>mask</i>	A mask indicating which methods within 'listener' should be invoked.

See also

DDS::PARTICIPANT_QOS_DEFAULT

[DomainParticipantListener](#)

DDS for a list of STATUS constants available to create the mask

2.6.2.2 ReturnCode_t delete_participant (DomainParticipant participant)

Destroy a [DomainParticipant](#). This routine will fail if all Entities (Publishers, Subscribers, etc) created through this [DomainParticipant](#) have not yet been deleted. (In this case, RETCODE_PRECONDITION_NOT_MET will be returned.)

Parameters

<i>participant</i>	A DomainParticipant previously created by a call to create_participant()
--------------------	--

2.6.2.3 ReturnCode_t get_default_participant_qos (DomainParticipantQos qos)

Provides access to the default Participant qos held in the factory. The provided **qos** argument is populated with the default qos settings.

2.6.2.4 static DomainParticipantFactory get_instance () [static]

Get access to the singleton [DomainParticipantFactory](#).

2.6.2.5 ReturnCode_t get_qos (DomainParticipantFactoryQos qos)

Gets the current QoS settings of the [DomainParticipantFactory](#).

2.6.2.6 DomainParticipant lookup_participant (long domain_id)

Find a [DomainParticipant](#) with the provided domain_id. If there are multiple participants in existence with the specified id, then one of them will be returned. (Which one is not specified.)

2.6.2.7 ReturnCode_t set_default_participant_qos (DomainParticipantQos qos)

Sets the default [DomainParticipantQos](#) held in the factory. This default qos will be used during subsequent calls to [DomainParticipantFactory.create_participant\(\)](#) if the special DDS.PARTICIPANT_QOS_DEFAULT value is provided for **qos**.

This routine may fail if the provided **qos** argument is not internally consistent. In this case, `ReturnCode_t.RETCODE_INCONSISTENT_POLICY` will be returned, and no changes will be made to the [DomainParticipantFactory](#).

2.6.2.8 `ReturnCode_t set_qos ( DomainParticipantFactoryQos qos )`

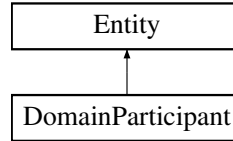
Sets the [DomainParticipantFactory](#) QoS values. These QoS values affect the behavior of the factory.

This routine may fail if the provided **qos** argument is not internally consistent. In this case, `ReturnCode_t.RETCODE_INCONSISTENT_POLICY` will be returned, and no changes will be made to the [DomainParticipantFactory](#).

2.7 DomainParticipant Class Reference

The [DomainParticipant](#) is used to configure, create and destroy [Publisher](#), [Subscriber](#) and [Topic](#) objects.

Inheritance diagram for DomainParticipant:



Public Member Functions

- `ReturnCode_t enable ()`
Enables the [DomainParticipant](#).
- `ReturnCode_t get_qos (DomainParticipantQos qos)`
- `ReturnCode_t set_qos (DomainParticipantQos qos)`
- `ReturnCode_t set_listener (DomainParticipantListener new_listener, long mask)`
- `DomainParticipantListener get_listener ()`
- `Publisher create_publisher (PublisherQos qos, PublisherListener listener, long mask)`
- `ReturnCode_t delete_publisher (Publisher p)`
- `Subscriber create_subscriber (SubscriberQos qos, SubscriberListener listener, long mask)`
- `ReturnCode_t delete_subscriber (Subscriber s)`
- `Topic create_topic (String topic_name, String type_name, TopicQos qos, TopicListener listener, long mask)`
- `ContentFilteredTopic create_contentfilteredtopic (String name, Topic related_topic, String filter_expression, Vector filter_parameters)`
- `ReturnCode_t delete_contentfilteredtopic (ContentFilteredTopic cft)`
- `MultiTopic create_multitopic (String name, String type_name, String subscription_expression, Vector expression_params)`
- `ReturnCode_t delete_multitopic (MultiTopic a_multitopic)`
- `ReturnCode_t delete_topic (Topic topic)`
- `Topic find_topic (String topic_name, Duration_t timeout)`
- `TopicDescription lookup_topicdescription (String name)`
- `Subscriber get_builtin_subscriber ()`
- `ReturnCode_t ignore_participant (InstanceHandle_t handle)`
- `ReturnCode_t ignore_topic (InstanceHandle_t handle)`
- `ReturnCode_t ignore_publication (InstanceHandle_t handle)`
- `ReturnCode_t ignore_subscription (InstanceHandle_t handle)`
- `long get_domain_id ()`
- `ReturnCode_t delete_contained_entities ()`
- `ReturnCode_t assert_liveliness ()`
- `ReturnCode_t set_default_publisher_qos (PublisherQos qos)`
- `ReturnCode_t get_default_publisher_qos (PublisherQos qos)`
- `ReturnCode_t set_default_subscriber_qos (SubscriberQos qos)`
- `ReturnCode_t get_default_subscriber_qos (SubscriberQos qos)`
- `ReturnCode_t set_default_topic_qos (TopicQos qos)`
- `ReturnCode_t get_default_topic_qos (TopicQos qos)`
- `ReturnCode_t get_discovered_participants (InstanceHandleSeq participant_handles)`

- `ReturnCode_t get_discovered_participant_data` (`ParticipantBuiltinTopicData` participant_data, `InstanceHandle_t` participant_handle)
- `ReturnCode_t get_discovered_topics` (Vector topic_handles)
- `ReturnCode_t get_discovered_topic_data` (`TopicBuiltinTopicData` topic_data, `InstanceHandle_t` topic_handle)
- `boolean contains_entity` (`InstanceHandle_t` handle)
- `ReturnCode_t get_current_time` (`Time_t` current_time)
- `ReturnCode_t register_type` (`TypeSupport` ts, String type_name)
Registers a `TypeSupport` with the Participant.
- `ReturnCode_t get_guid` (`GUID_t` g)
Access the GUID which uniquely identifies this participant.
- `ReturnCode_t add_tcp_transport` ()
Removed. Install the TCP `Transport`. Must be called before enabling the `DomainParticipant`. The `DomainParticipant` will default to using standard UDP RTPS transport. Adding the TCP `Transport` will override this default.
- `ReturnCode_t add_transport` (`Transport` t)
Install the provided `Transport`.
- `ReturnCode_t do_work` (`Duration_t` time_slice)
Transfer control to the `DomainParticipant` so it can do background processing.
- `ReturnCode_t builtin_wait_for_acknowledgments` (`Duration_t` max_wait)

2.7.1 Detailed Description

The `DomainParticipant` is used to configure, create and destroy `Publisher`, `Subscriber` and `Topic` objects.

The `DomainParticipant` is a container for the objects that it creates. The objects operate within the **domain** identified by the **domain_id** provided when the `DomainParticipant` was created. Objects within different **domains** do not communicate or interfere with each other.

2.7.2 Member Function Documentation

2.7.2.1 `ReturnCode_t add_tcp_transport ( )`

Removed. Install the TCP `Transport`. Must be called before enabling the `DomainParticipant`. The `DomainParticipant` will default to using standard UDP RTPS transport. Adding the TCP `Transport` will override this default.

Deprecated

See also

`DomainParticipant::add_transport(Transport)` `add_transport()`
`TcpTransportConfig`
`TcpTransport`
`UdpTransportConfig`
`UdpTransport`

2.7.2.2 `ReturnCode_t add_transport ( Transport t )`

Install the provided `Transport`.

Must be called before enabling the `DomainParticipant`. The `DomainParticipant` will default to using standard UDP RTPS transport, if no other transports are added.

See also

[LmtTransportConfig](#)
[LmtTransport](#)
[TcpTransportConfig](#)
[TcpTransport](#)
[UdpTransportConfig](#)
[UdpTransport](#)

2.7.2.3 ReturnCode_t assert_liveliness ()

This operation manually asserts the liveliness of the [DomainParticipant dp](#). This operation indicates that the [DomainParticipant](#) is still alive. It is effective only if the [DomainParticipant](#) contains one or more [DataWriter](#) objects with LIVELINESS QoS set to MANUAL_BY_PARTICIPANT_LIVELINESS_QOS. In this case, the operation will assert the liveliness of those particular [DataWriter](#) objects.

Writing data via the [DataWriter write](#) operation automatically asserts the liveliness of the [DataWriter](#) and its containing [DomainParticipant](#). Therefore, the use of [DomainParticipant.assert_liveliness\(\)](#) is needed only if the application is not writing data frequently enough to keep the [DataWriter](#) considered 'alive'.

2.7.2.4 ReturnCode_t builtin_wait_for_acknowledgments (Duration_t max_wait)

Ensure that local discovery information has been received by all known peers.

2.7.2.5 boolean contains_entity (InstanceHandle_t handle)

This operation checks whether or not the given handle **a_handle** represents an object that was created by the [DomainParticipant d](#). This applies recursively to all objects created by the [DomainParticipant](#).

The routine will return true (non-zero) if the handle is found, zero otherwise.

2.7.2.6 ContentFilteredTopic create_contentfilteredtopic (String name, Topic related_topic, String filter_expression, Vector filter_parameters)

This operation creates a [ContentFilteredTopic](#).

The [ContentFilteredTopic](#) is associated with another un-filtered topic **related_topic**.

The **filter_expression** is an SQL like condition expression, and **filter_parameters** provide optional parameters that are referenced by the **filter_expression**. The syntax of the filter expression is similar to the WHERE clause in SQL. For example "x<4" is a valid filter expression. It would test that data member 'x' is less than value '4'. If the filter expression evaluates to TRUE, then the data sample will be available, otherwise the data sample would be 'filtered' (excluded).

The filter_expression can refer to parameters. The syntax for parameters is the percent sign "%" followed by a number. The number is the index of the parameter in the **filter_parameters** sequence. Parameters are counted starting at zero. So, "%0" refers to the first parameter, and "%4" refers to the fifth parameter. Using this syntax, the expression "x<%0" would test the value of 'x' against the first parameter in the sequence.

2.7.2.7 MultiTopic create_multitopic (String name, String type_name, String subscription_expression, Vector expression_params)

Not Yet Supported This is currently unsupported in the Java language binding.

2.7.2.8 Publisher `create_publisher ( PublisherQos qos, PublisherListener listener, long mask )`

This operation creates a [Publisher](#) with the specified PublisherQoS settings and [PublisherListener](#). The value **DDS.PUBLISHER_QOS_DEFAULT** may be provided for the **qos** argument. This will indicate that the the default publisher QoS settings held in the [DomainParticipant](#) should be used.

This operation may fail and return NULL if the QoS settings are internally inconsistent.

See also

DDS for a list of STATUS constants available to create the mask

2.7.2.9 Subscriber `create_subscriber ( SubscriberQos qos, SubscriberListener listener, long mask )`

This operation creates a [Subscriber](#) with the specified [SubscriberQos](#) and [SubscriberListener](#). The value **DDS.SUBSCRIBER_QOS_DEFAULT** may be provided for the **qos** argument. This will indicate that the the default subscriber QoS settings held in the [DomainParticipant](#) should be used.

This operation may fail and return NULL if the QoS settings are internally inconsistent.

See also

DDS for a list of STATUS constants available to create the mask

2.7.2.10 Topic `create_topic ( String topic_name, String type_name, TopicQos qos, TopicListener listener, long mask )`

This operation creates a [Topic](#) with the specified [TopicQos](#) settings and [TopicListener](#).

The value **DDS.TOPIC_QOS_DEFAULT** may be provided for the **qos** argument. This will indicate that the the default topic QoS settings held in the [DomainParticipant](#) should be used.

The topic is created with a reference to the data type indicated by the **type_name** argument. The data type must have been registered with the [DomainParticipant](#) (by a call to [register_type\(\)](#) on the appropriate [TypeSupport](#) object) prior to calling [create_topic\(\)](#).

This operation may fail and return NULL if the QoS settings are internally inconsistent.

The topic returned from this operation (if not NULL) must be deleted by calling [DomainParticipant.delete_topic\(\)](#).

See also

DDS for a list of STATUS constants available to create the mask

2.7.2.11 `ReturnCode_t delete_contained_entities ( )`

This operation deletes all the objects created by means of the **create** operations on [DomainParticipant dp](#). This routine will recursively call the corresponding [delete_contained_entities\(\)](#) operation on each of the contained objects (Publishers, Subscribers, Topics, ContentFilteredTopics, and MultiTopics). If successful, this operation will recursively delete all objects contained with this [DomainParticipant](#). After successful execution, the application may delete the [DomainParticipant](#) by calling [DomainParticipantFactory.delete_participant\(\)](#).

If any of the objects cannot be deleted, this routine will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`.

2.7.2.12 `ReturnCode_t delete_contentfilteredtopic ( ContentFilteredTopic cft )`

This operation deletes a previously allocated [ContentFilteredTopic](#). This operation will fail if there are any [DataReader](#), [DataWriter](#), [ContentFilteredTopic](#), or [MultiTopic](#) objects that reference the specified [Topic](#). In this case, the operation will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`.

The [Topic](#) must be owned by DomainParticipant **dp**; otherwise `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET` will be returned.

2.7.2.13 `ReturnCode_t delete_multitopic ( MultiTopic a_multitopic )`

Not Yet Supported This is currently unsupported in the Java language binding.

2.7.2.14 `ReturnCode_t delete_publisher ( Publisher p )`

This operation deletes an existing [Publisher](#). A [Publisher](#) cannot be deleted if it contains any [DataWriter](#) objects. In this case, **delete_publisher** will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`.

If the [DomainParticipant](#) **dp** does not contain the provided [Publisher](#) **p**, the operation will return **ReturnCode_t.RETCODE_PRECONDITION_NOT_MET**.

2.7.2.15 `ReturnCode_t delete_subscriber ( Subscriber s )`

This operation deletes an existing [Subscriber](#). A [Subscriber](#) cannot be deleted if it contains any [DataReader](#) objects. In this case, **delete_subscriber** will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`.

If the [DomainParticipant](#) **dp** does not contain the provided [Subscriber](#) **s**, the operation will return **ReturnCode_t.RETCODE_PRECONDITION_NOT_MET**.

2.7.2.16 `ReturnCode_t delete_topic ( Topic topic )`

This operation deletes a [Topic](#). This operation will fail if there are any [DataReader](#), [DataWriter](#), [ContentFilteredTopic](#), or [MultiTopic](#) objects that reference the specified [Topic](#). In this case, the operation will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`.

The [Topic](#) must be owned by DomainParticipant **dp**; otherwise `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET` will be returned.

2.7.2.17 `ReturnCode_t do_work ( Duration_t time_slice )`

Transfer control to the [DomainParticipant](#) so it can do background processing.

This function is to be called only if the [DomainParticipant](#) is configured to use NO threads via the `thread_model` QoS policy. The participant will continue to execute background processing until at least 'time_slice' time passes. This routine will attempt to use no more than 'time_slice' time, but this is not a guarantee.

Note

If the [DomainParticipant](#) has threads enabled (default case), then the Participant takes care of scheduling background work on its own, and this routine is not necessary and should not be called.

2.7.2.18 `ReturnCode_t enable ( )`

Enables the [DomainParticipant](#).

A [DomainParticipant](#) is created either enabled or not based on the `DomainParticipantFactoryQoS` setting **entity_factory**. When a [DomainParticipant](#) is not enabled, only the following sub-set of all [DomainParticipant](#) operations are legal:

- operations to get and set QoS policies,
- factory operations (create, delete),
- [get_statuscondition\(\)](#),
- [get_status_changes\(\)](#),
- lookup operations

Any other [DomainParticipant](#) operation will return the `ReturnCode_t.RETCODE_NOT_ENABLED` error. [DomainParticipant.enable\(\)](#) may be called on an already enabled [DomainParticipant](#) [it will have no effect].

2.7.2.19 `Topic find_topic ( String topic_name, Duration_t timeout )`

This operation returns a [Topic](#) identified by the provided **topic_name**. If a topic with name **topic_name** is known to exist, the function returns this matching topic immediately. Otherwise, the operation will block for up to the **timeout** duration. If a topic with name **topic_name** is discovered or otherwise created, the operation will cease to wait, and will return the matching topic.

If a matching topic is not known by the time the **timeout** has expired, the operation will return NULL.

Like `DomainParticipant_create_topic()`, the topic returned from this operation (if not NULL) must be deleted by calling [DomainParticipant.delete_topic\(\)](#).

2.7.2.20 `Subscriber get_builtin_subscriber ( )`

Returns the built-in [Subscriber](#).

Each [DomainParticipant](#) contains several built-in [Topic](#) objects as well as corresponding [DataReader](#) objects to access them. These built-in [DataReader](#) objects belong to the single built-in [Subscriber](#) that is accessed through this operation.

The built-in Topics are used to communicate information about other [DomainParticipant](#), [Topic](#), [DataReader](#), and [DataWriter](#) objects.

An application should not explicitly delete the [Subscriber](#) returned by this operation - it is managed internally.

See also

[ParticipantBuiltinTopicDataDataReader](#)
[PublicationBuiltinTopicDataDataReader](#)
[SubscriptionBuiltinTopicDataDataReader](#)

2.7.2.21 `ReturnCode_t get_current_time ( Time_t current_time )`

This operation returns the current value of the time used by the service.

2.7.2.22 `ReturnCode_t get_default_publisher_qos ( PublisherQos qos )`

Provides access to the default [PublisherQos](#) settings held in the factory. The provided **qos** argument is populated with the default qos settings.

2.7.2.23 `ReturnCode_t get_default_subscriber_qos ( SubscriberQos qos )`

Provides access to the default [SubscriberQos](#) settings held in the factory. The provided **qos** argument is populated with the default qos settings.

2.7.2.24 `ReturnCode_t get_default_topic_qos ( TopicQos qos )`

Provides access to the default [TopicQos](#) settings held in the factory. The provided **qos** argument is populated with the default qos settings.

2.7.2.25 `ReturnCode_t get_discovered_participant_data ( ParticipantBuiltinTopicData participant_data, InstanceHandle_t participant_handle )`

This operation returns data that describes a particular discovered participant identified by **participant_handle**. An appropriate handle can be obtained through a call to [DomainParticipant.get_discovered_participants\(\)](#).

If **participant_handle** does not identify a known [DomainParticipant](#), this routine will return `ReturnCode_t.RETCODE_PRECONDITION_N`.

2.7.2.26 `ReturnCode_t get_discovered_participants ( InstanceHandleSeq participant_handles )`

This operation returns the list of handles identifying the [DomainParticipant](#) objects that have been discovered. The returned list will include only participants which are in the same domain as participant **d**, and which are not explicitly ignored as a result of a call to [DomainParticipant.ignore_participant\(\)](#).

2.7.2.27 `ReturnCode_t get_discovered_topic_data ( TopicBuiltinTopicData topic_data, InstanceHandle_t topic_handle )`

This operation returns data that describes a particular discovered topic identified by **topic_handle**. An appropriate handle can be obtained through a call to [DomainParticipant.get_discovered_topics\(\)](#).

If **topic_handle** does not identify a known [Topic](#), this routine will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`.

Not Yet Supported This is currently unsupported in the Java language binding.

2.7.2.28 `ReturnCode_t get_discovered_topics ( Vector topic_handles )`

This operation returns the list of handles identifying the [Topic](#) objects that have been discovered. The returned list will include only topics which are in the same domain as participant **d**, and which are not explicitly ignored as a result of a call to [DomainParticipant.ignore_topic\(\)](#).

Not Yet Supported This is currently unsupported in the Java language binding.

2.7.2.29 `long get_domain_id ( )`

Gets the **domain_id** to which the [DomainParticipant](#) belongs.

2.7.2.30 `DomainParticipantListener get_listener ( )`

This operation returns the currently installed [DomainParticipantListener](#). Because the infrastructure makes a copy of the listener provided in [DomainParticipant.set_listener\(\)](#), the returned structure pointer will not match the pointer originally provided. However, the function pointers within the structure will match. Also, the application should not free the data referenced by the returned pointer.

2.7.2.31 `ReturnCode_t get_qos ( DomainParticipantQos qos )`

Gets the DomainParticipantQoS settings of the [DomainParticipant](#).

2.7.2.32 `ReturnCode_t ignore_participant ( InstanceHandle_t handle )`

Instructs the [DomainParticipant](#) **dp** to ignore the external [DomainParticipant](#) identified by **handle**. This will cause the infrastructure to behave as if the external participant **handle** did not exist. The handle can be discovered by accessing the built-in topic **DCSParticipant** via the appropriate built-in [DataReader](#).

There is no mechanism to reverse this operation.

Not Yet Supported This is currently unsupported in the Java language binding.

2.7.2.33 `ReturnCode_t ignore_publication ( InstanceHandle_t handle )`

This operation instructs the [DomainParticipant](#) **dp** to ignore a Publication identified by **handle**. The handle can be discovered by accessing the built-in topic **DCSPublication** via the appropriate built-in [DataReader](#).

There is no mechanism to reverse this operation.

Not Yet Supported This is currently unsupported in the Java language binding.

2.7.2.34 `ReturnCode_t ignore_subscription ( InstanceHandle_t handle )`

This operation instructs the [DomainParticipant](#) **dp** to ignore a Subscription identified by **handle**. The handle can be discovered by accessing the built-in topic **DCPSSubscription** via the appropriate built-in [DataReader](#).

There is no mechanism to reverse this operation.

Not Yet Supported This is currently unsupported in the Java language binding.

2.7.2.35 `ReturnCode_t ignore_topic ( InstanceHandle_t handle )`

This operation instructs the [DomainParticipant](#) **dp** to ignore a [Topic](#) identified by **handle**. This can be used to save resources if a participant will never participate on certain Topics. The handle can be discovered by accessing the built-in topic **DCPSTopic** via the appropriate built-in [DataReader](#).

There is no mechanism to reverse this operation.

Not Yet Supported This is currently unsupported in the Java language binding.

2.7.2.36 TopicDescription lookup_topicdescription (String *name*)

This operation returns an existing, locally-created [TopicDescription](#), named **name**. If a [TopicDescription](#) named **name** does not exist, then this routine will return NULL.

2.7.2.37 ReturnCode_t register_type (TypeSupport *ts*, String *type_name*)

Registers a [TypeSupport](#) with the Participant.

Associates a [TypeSupport](#) object with the provided 'type_name'. This [TypeSupport](#) will be used to handle data that has a matching type_name. If a [TypeSupport](#) has already been registered for the provided type_name, and the new type does not match the previously registered type, then an error is returned and the previously registered [TypeSupport](#) is maintained. [Upon error, the new type is not registered.]

2.7.2.38 ReturnCode_t set_default_publisher_qos (PublisherQos *qos*)

Sets the default [PublisherQos](#) held in the [DomainParticipant](#). This default qos will be used during subsequent calls to [DomainParticipant.create_publisher\(\)](#) if the special DDS.PUBLISHER_QOS_DEFAULT value is provided for **qos**. This routine may fail if the provided **qos** argument is not internally consistent. In this case, ReturnCode_t.RETCODE_INCONSISTENT_POLICY will be returned, and no changes will be made to the [DomainParticipant](#).

2.7.2.39 ReturnCode_t set_default_subscriber_qos (SubscriberQos *qos*)

Sets the default [SubscriberQos](#) held in the [DomainParticipant](#). This default qos will be used during subsequent calls to [DomainParticipant.create_subscriber\(\)](#) if the special DDS.SUBSCRIBER_QOS_DEFAULT value is provided for **qos**. This routine may fail if the provided **qos** argument is not internally consistent. In this case, ReturnCode_t.RETCODE_INCONSISTENT_POLICY will be returned, and no changes will be made to the [DomainParticipant](#).

2.7.2.40 ReturnCode_t set_default_topic_qos (TopicQos *qos*)

Sets the default [TopicQos](#) held in the [DomainParticipant](#). This default qos will be used during subsequent calls to [DomainParticipant.create_topic\(\)](#) [and related] if the special DDS.TOPIC_QOS_DEFAULT value is provided for **qos**. This routine may fail if the provided **qos** argument is not internally consistent. In this case, ReturnCode_t.RETCODE_INCONSISTENT_POLICY will be returned, and no changes will be made to the [DomainParticipant](#).

2.7.2.41 ReturnCode_t set_listener (DomainParticipantListener *new_listener*, long *mask*)

This operation installs a [DomainParticipantListener](#) on the [DomainParticipant](#). Only one listener may be attached to a [DomainParticipant](#) at a time. A call to [set_listener\(\)](#) will replace any current listener with **a_listener**.

a_listener can be NULL, which indicates a listener that does nothing.

The infrastructure will make an internal copy of the listener structure so that it need not be persisted by the application.

See also

DDS for a list of available STATUS constants which may be or'ed together to construct the mask.

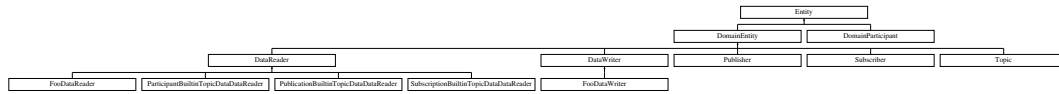
2.7.2.42 ReturnCode_t set_qos (DomainParticipantQos qos)

Sets the DomainParticipantQoS values. These QoS values affect the behavior of the [DomainParticipant](#). This routine may fail if the provided **qos** argument is not internally consistent. In this case, ReturnCode_t.RETCODE_INCONSISTENT_POLICY will be returned, and no changes will be made to the [DomainParticipant](#) QoS.

2.8 Entity Class Reference

Base class for all DDS Entities.

Inheritance diagram for Entity:



Public Member Functions

- [StatusCondition get_statuscondition \(\)](#)
- [int get_status_changes \(\)](#)
- [ReturnCode_t enable \(\)](#)
- [InstanceHandle_t get_instance_handle \(\)](#)

2.8.1 Detailed Description

Base class for all DDS Entities.

2.8.2 Member Function Documentation

2.8.2.1 ReturnCode_t enable ()

Enable this [Entity](#). An [Entity](#) will begin to communicate only after it is enabled.

2.8.2.2 InstanceHandle_t get_instance_handle ()

Gets the handle that locally identifies this [Entity](#).

2.8.2.3 int get_status_changes ()

Gets the current status changes from this [Entity](#).

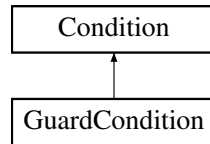
2.8.2.4 StatusCondition get_statuscondition ()

Gets the [StatusCondition](#) associated with this [Entity](#).

2.9 GuardCondition Class Reference

A [GuardCondition](#) is a [Condition](#) where the **trigger_value** is under application control.

Inheritance diagram for GuardCondition:



Public Member Functions

- [GuardCondition](#) ()

2.9.1 Detailed Description

A [GuardCondition](#) is a [Condition](#) where the **trigger_value** is under application control.

2.9.2 Constructor & Destructor Documentation

2.9.2.1 [GuardCondition](#) ()

This routine creates a [GuardCondition](#).

2.10 Locator Class Reference

Network address.

Public Attributes

- String [addr](#)

2.10.1 Detailed Description

Network address.

For a list of supported LOCATOR_KINDs, refer to `include/dds/dds.h`

2.10.2 Member Data Documentation

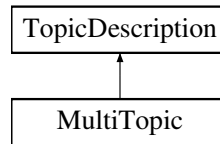
2.10.2.1 String `addr`

an IP address will look like: "10.0.1.123"

2.11 MultiTopic Class Reference

[MultiTopic](#) provides a topic that may include data from multiple Topics.

Inheritance diagram for MultiTopic:



Public Member Functions

- [DomainParticipant](#) [get_participant](#) ()
- String [get_type_name](#) ()
- String [get_name](#) ()

2.11.1 Detailed Description

[MultiTopic](#) provides a topic that may include data from multiple Topics.

Not Yet Supported This is currently unsupported in the Java language binding.

2.11.2 Member Function Documentation

2.11.2.1 String [get_name](#) ()

Not Yet Supported This routine is not yet implemented.

Implements [TopicDescription](#).

2.11.2.2 DomainParticipant [get_participant](#) ()

Not Yet Supported This routine is not yet implemented.

Implements [TopicDescription](#).

2.11.2.3 String [get_type_name](#) ()

Not Yet Supported This routine is not yet implemented.

Implements [TopicDescription](#).

2.12 ParticipantLocator Class Reference

Describes a the location and identity of a potential peer [DomainParticipant](#).

Public Attributes

- long [participant_id](#)
- long [participant_id_max](#)
- [Locator](#) [participant_locator](#)

2.12.1 Detailed Description

Describes a the location and identity of a potential peer [DomainParticipant](#).

2.12.2 Member Data Documentation

2.12.2.1 long participant_id

one-up unique [DomainParticipant](#) identifier within an application

2.12.2.2 long participant_id_max

if non-zero, we will use a range of ids from participant_id to participant_id_max

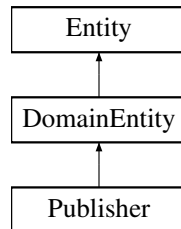
2.12.2.3 Locator participant_locator

newtork address

2.13 Publisher Class Reference

The [Publisher](#) configures, creates, manages and destroys DataWriters.

Inheritance diagram for Publisher:



Public Member Functions

- `ReturnCode_t enable ()`
- `DomainParticipant get_participant ()`
- `DataWriter create_datawriter (Topic topic, DataWriterQos qos, DataWriterListener listener, long mask)`
- `ReturnCode_t delete_datawriter (DataWriter datawriter)`
- `ReturnCode_t delete_contained_entities ()`
- `DataWriter lookup_datawriter (String topic_name)`
- `ReturnCode_t set_qos (PublisherQos qos)`
- `ReturnCode_t get_qos (PublisherQos qos)`
- `ReturnCode_t set_listener (PublisherListener new_listener, long mask)`
- `PublisherListener get_listener ()`
- `ReturnCode_t suspend_publications ()`
- `ReturnCode_t resume_publications ()`
- `ReturnCode_t begin_coherent_changes ()`
- `ReturnCode_t end_coherent_changes ()`
- `ReturnCode_t wait_for_acknowledgments (Duration_t max_wait)`
Block until all writers contained by this publisher have received acknowledgements.
- `ReturnCode_t set_default_datawriter_qos (DataWriterQos qos)`
- `ReturnCode_t get_default_datawriter_qos (DataWriterQos qos)`
- `ReturnCode_t copy_from_topic_qos (DataWriterQos qos, TopicQos topic_qos)`

2.13.1 Detailed Description

The [Publisher](#) configures, creates, manages and destroys DataWriters.

2.13.2 Member Function Documentation

2.13.2.1 `ReturnCode_t begin_coherent_changes ( )`

Not Yet Supported This operation is not yet implemented.

2.13.2.2 `ReturnCode_t copy_from_topic_qos ( DataWriterQos qos, TopicQos topic\_qos )`

This operation copies the QoS settings in [a_topic_qos](#) to the corresponding settings in [a_datawriter_qos](#). The [a_datawriter_qos](#) parameter is populated with a copy of the QoS policies from the [a_topic_qos](#) structure. QoS entries in the datawriter qos structure will be overwritten with the values from the topic.

2.13.2.3 `DataWriter create_datawriter ( Topic topic, DataWriterQos qos, DataWriterListener listener, long mask )`

This operation creates a [DataWriter](#). The created [DataWriter](#) is contained within the [Publisher p](#). It is associated with the [Topic](#), [ContentFilteredTopic](#), or [MultiTopic](#) indicated by [a_topic](#), and has the [DataWriterQos](#) indicated by [qos](#). The [qos](#) argument may be passed `DDS.DATAWRITER_QOS_DEFAULT`, which indicates that the [Publisher](#) should use its currently configured default data writer QoS values. The [DataWriterListener a_listener](#), is installed at creation time.

The created [DataWriter](#) (if not NULL) must be destroyed by a call to [Publisher.delete_datawriter\(\)](#).

This routine will fail if the provided QoS settings are internally inconsistent. In this case, the routine will return **NULL**.

See also

DDS for a list of STATUS constants available to create the mask

2.13.2.4 `ReturnCode_t delete_contained_entities ( )`

This operation deletes all the DataWriters created by means of the [Publisher.create_datawriter\(\)](#) operation on the [Publisher p](#). This routine will recursively call the corresponding [delete_contained_entities\(\)](#) operation on each of the contained [DataWriter](#) objects. After successful execution, the application may delete the [Publisher](#) by calling [DomainParticipant.delete_publisher\(\)](#).

If any of the objects cannot be deleted, this routine will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`.

2.13.2.5 `ReturnCode_t delete_datawriter ( DataWriter datawriter )`

This operation deletes a [DataWriter](#). If the provided [DataWriter a_datawriter](#) was not created by [Publisher p](#), the routine will fail and will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`.

2.13.2.6 `ReturnCode_t enable ( )`

Enables the [Publisher](#). A [Publisher](#) is created either enabled or not based on the [DomainParticipantQos](#) setting **entity_factory**. When a [Publisher](#) is not enabled, only the following sub-set of all [Publisher](#) operations are legal:

- operations to get and set QoS policies,
- factory operations (create, delete),
- [get_statuscondition\(\)](#),
- [get_status_changes\(\)](#),
- lookup operations

Any other operation may return the `ReturnCode_t.RETCODE_NOT_ENABLED` error. [Publisher_enable\(\)](#) may be called on an already enabled [Subscriber](#) [it will have no effect].

2.13.2.7 `ReturnCode_t end_coherent_changes ( )`

Not Yet Supported This operation is not yet implemented.

2.13.2.8 `ReturnCode_t get_default_datawriter_qos ( DataWriterQos qos )`

Provides access to the default [DataWriterQos](#) settings held in the [Publisher p](#). The provided **qos** argument is populated with the current default qos settings.

2.13.2.9 `PublisherListener get_listener ( )`

This operation returns the currently installed [PublisherListener](#).

2.13.2.10 `DomainParticipant get_participant ( )`

This operation returns the [DomainParticipant](#) this [Publisher](#) belongs to.

2.13.2.11 `ReturnCode_t get_qos ( PublisherQos qos )`

Returns the current [PublisherQos](#) settings held in the [Publisher p](#). The **qos** parameter is populated with a copy of the current [Publisher](#) QoS properties.

2.13.2.12 `DataWriter lookup_datawriter ( String topic_name )`

This operation retrieves a previously-created [DataWriter](#) contained in the [Publisher](#), attached to a [Topic](#) named **topic_name**. If multiple [DataWriters](#) are found, one of them will be returned. If no matching [DataWriter](#) is found, this routine will return **NULL**.

2.13.2.13 `ReturnCode_t resume_publications ( )`

Not Yet Supported This operation is not yet implemented.

2.13.2.14 `ReturnCode_t set_default_datawriter_qos ( DataWriterQos qos )`

Sets the default [DataWriterQos](#) held in the [Publisher](#). This default qos will be used during subsequent calls to [Publisher.create_datawriter\(\)](#) if the special `DDS.DATAWRITER_QOS_DEFAULT` value is provided for **qos**. This routine may fail if the provided **qos** argument is not internally consistent. In this case, `ReturnCode_t.RETCODE_INCONSISTENT_POLICY` will be returned, and no changes will be made to the [Publisher](#).

2.13.2.15 `ReturnCode_t set_listener ( PublisherListener new_listener, long mask )`

Installs a [PublisherListener](#) on [Publisher p](#). Only one listener may be attached to a [Publisher](#) at a time. A call to [set_listener\(\)](#) will replace any current listener with **a_listener**.

a_listener can be **NULL**, which indicates a listener that does nothing.

See also

DDS for a list of available STATUS constants which may be or'ed together to construct the mask.

2.13.2.16 ReturnCode_t set_qos (PublisherQos qos)

Sets the [PublisherQos](#) values. These QoS values affect the behavior of the [Publisher](#). This routine may fail if the provided **qos** argument is not internally consistent. In this case, ReturnCode_t.RETCODE_INCONSISTENT_POLICY will be returned, and no changes will be made to the [Publisher](#) QoS.

2.13.2.17 ReturnCode_t suspend_publications ()

Not Yet Supported This operation is not yet implemented.

2.13.2.18 ReturnCode_t wait_for_acknowledgments (Duration_t max_wait)

Block until all writers contained by this publisher have received acknowledgements.

This routine will block until all data written by contained writers has been acknowledged, or until the 'max_wait' duration has passed. 'max_wait' can be set to INFINITE, in which case this routine may block indefinitely.

Return values

<i>DDS_RETCODE_TIME_OUT</i>	returned if 'max_wait' passes before all acks are received
<i>DDS_RETCODE_OK</i>	returned if all acks have been received before 'max_wait'

2.14 QosPolicyCount Class Reference

A QoS Policy identifier, along with a count.

2.14.1 Detailed Description

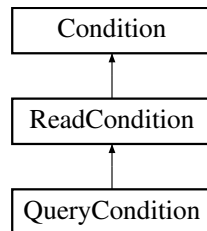
A QoS Policy identifier, along with a count.

This class is used with the [OfferedIncompatibleQosStatus](#) and [RequestedIncompatibleQosStatus](#) classes to describe the incompatibility between [DataReader](#) and [DataWriter](#) Qos configuration.

2.15 QueryCondition Class Reference

The **trigger_value** is driven by the data available, after applying the filter, in the associated [DataReader](#).

Inheritance diagram for QueryCondition:



Public Member Functions

- String [get_query_expression](#) ()
Provides access to the query expression.
- Vector< String > [get_query_parameters](#) ()
Provides access to the parameters of the query expression.
- ReturnCode_t [set_query_parameters](#) (Vector parameters)
Modifies the parameters of the query expression.

2.15.1 Detailed Description

The **trigger_value** is driven by the data available, after applying the filter, in the associated [DataReader](#).

CoreDX DDS fully supports QueryConditions as an argument to `DataReader::read_w_condition()` and `DataReader::take_w_conditions()`

See also

[DataReader::create_querycondition\(\)](#)
[FooDataReader::read_w_condition\(\)](#)
[FooDataReader::take_w_condition\(\)](#)

Not Yet Supported CoreDX DDS does not yet support QueryConditions as triggers for a [WaitSet](#).

2.15.2 Member Function Documentation

2.15.2.1 String [get_query_expression](#) ()

Provides access to the query expression.

The query expression is an SQL syntax conditional expression that is provided when the [QueryCondition](#) is created.

See also

[DataReader::create_querycondition\(\)](#)

2.15.2.2 `Vector<String> get_query_parameters ( )`

Provides access to the parameters of the query expression.

The query expression is an SQL syntax conditional expression that is provided when the [QueryCondition](#) is created. The query expression may contain references to positional parameters of the form '%0', '%1'. These parameters can be changed dynamically to affect the expression.

See also

[DataReader::create_querycondition\(\)](#)
[QueryCondition::set_query_parameters\(\)](#)

2.15.2.3 `ReturnCode_t set_query_parameters ( Vector parameters )`

Modifies the parameters of the query expression.

The **query_expression** is an SQL like condition expression, and the **parameters** argument provides optional parameters that are referenced by the **query_expression**. The syntax for referring to parameters in a query_expression is the percent sign " " followed by a number. The number is the index of the parameter in the **query_parameters** sequence. Parameters are counted starting at zero. So, "%0" refers to the first parameter, and "%4" refers to the fifth parameter. Using this syntax, the expression "x<%0" would test the value of 'x' against the first parameter in the sequence.

This routine allows the parameters to be changed dynamically.

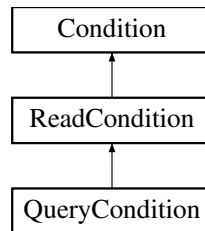
See also

[DataReader::create_querycondition\(\)](#)

2.16 ReadCondition Class Reference

A [ReadCondition](#) is a specialized [Condition](#) associated with a [DataReader](#).

Inheritance diagram for ReadCondition:



Public Member Functions

- [long get_sample_state_mask \(\)](#)
- [long get_view_state_mask \(\)](#)
- [long get_instance_state_mask \(\)](#)
- [DataReader get_datareader \(\)](#)

2.16.1 Detailed Description

A [ReadCondition](#) is a specialized [Condition](#) associated with a [DataReader](#).

The **trigger_value** is driven by the data available in the associated [DataReader](#). A [ReadCondition](#) is obtained by calling the `DataReader_create_readcondition()` function. When the [ReadCondition](#) is no longer needed, it should be destroyed by a call to `DataReader.delete_readcondition()`.

See also

```
DataReader::create\_readcondition\(long sample\_states, long view\_states, long instance\_states\)  
DataReader::delete\_readcondition\(ReadCondition\)
```

2.16.2 Member Function Documentation

2.16.2.1 [DataReader get_datareader \(\)](#)

Gets the single [DataReader](#) associated with this [ReadCondition](#).

2.16.2.2 [long get_instance_state_mask \(\)](#)

Gets the current value of the `instance_state_mask` in this [ReadCondition](#).

2.16.2.3 [long get_sample_state_mask \(\)](#)

Gets the current value of the `sample_state_mask` in this [ReadCondition](#).

2.16.2.4 long `get_view_state_mask ( )`

Gets the current value of the `view_state_mask` in this [ReadCondition](#).

2.17 SampleInfo Class Reference

The [SampleInfo](#) structure contains information associated with each Sample. The `DataReader.read()` and `take()` operations return two vectors. One vector contains Sample(s) and the other contains SampleInfo(s). There is a one-to-one correspondence between items in these two vectors. Each Sample is described by the corresponding [SampleInfo](#) instance.

Public Attributes

- long [sample_state](#)
- long [view_state](#)
- long [instance_state](#)
- [Time_t](#) [source_timestamp](#)
- [InstanceHandle_t](#) [instance_handle](#)
- [InstanceHandle_t](#) [publication_handle](#)
- int [disposed_generation_count](#)
- int [no_writers_generation_count](#)
- int [sample_rank](#)
- int [generation_rank](#)
- int [absolute_generation_rank](#)
- boolean [valid_data](#)

2.17.1 Detailed Description

The [SampleInfo](#) structure contains information associated with each Sample. The `DataReader.read()` and `take()` operations return two vectors. One vector contains Sample(s) and the other contains SampleInfo(s). There is a one-to-one correspondence between items in these two vectors. Each Sample is described by the corresponding [SampleInfo](#) instance.

2.17.2 Member Data Documentation

2.17.2.1 int [absolute_generation_rank](#)

The generation difference between this sample and the most recent sample. The **[absolute_generation_rank](#)** indicates the generation difference (ie, the number of times the instance was disposed and became alive again) between this sample, and the most recent sample (possibly not in the returned collection) of this instance.

2.17.2.2 int [disposed_generation_count](#)

The number of times the instance has become 'ALIVE' after being explicitly disposed. (Computed at the time the sample arrives at the [DataReader](#).)

2.17.2.3 int [generation_rank](#)

The generation difference of this sample and the most recent sample in the collection. **[generation_rank](#)** indicates the generation difference (ie, the number of times the instance was disposed and became alive again) between this sample, and the most recent sample in the collection related to this instance.

2.17.2.4 InstanceHandle_t instance_handle

The handle that locally identifies the associated instance.

2.17.2.5 long instance_state

Indicates whether the associated instance currently exists. **instance_state** can be one of:

DDS.ALIVE_INSTANCE_STATE

DDS.NOT_ALIVE_DISPOSED_INSTANCE_STATE

DDS.NOT_ALIVE_NO_WRITERS_INSTANCE_STATE

Can use DDS.NOT_ALIVE_INSTANCE_STATE as a test for either 'NOT_ALIVE' state. For example if (sample_info->view_state & DDS.NOT_ALIVE_INSTANCE_STATE) ...

2.17.2.6 int no_writers_generation_count

The number of times the instance has become 'ALIVE' after being automatically disposed due to no active writers. (Computed at the time the sample arrives at the [DataReader](#).)

2.17.2.7 InstanceHandle_t publication_handle

The local handle of the source [DataWriter](#).

2.17.2.8 int sample_rank

Number of samples related to this instances that follow in the collection returned by the [DataReader](#) **read** or **take** operations.

2.17.2.9 long sample_state

The associated data sample has/has not been read previously.

sample_state indicates whether or not the [DataReader](#) has previously read the associated sample.

One of:

DDS.READ_SAMPLE_STATE

DDS.NOT_READ_SAMPLE_STATE.

Use DDS.ANY_SAMPLE_STATE to test for either state.

2.17.2.10 Time_t source_timestamp

The time provided by the [DataWriter](#) when the sample was written.

2.17.2.11 boolean valid_data

Is set to true if the associated DataSample contains data. The associated DataSample may not contain data if it this sample indicates a change in sample state (for example ALIVE -> DISPOSED).

2.17.2.12 long view_state

Associated instance has/has not been seen before. **view_state** indicates whether the [DataReader](#) has already seen samples for the most current generation of the related instance.

view_state can be one of:

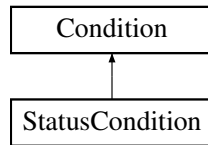
DDS.NEW_VIEW_STATE

DDS.NOT_NEW_VIEW_STATE

2.18 StatusCondition Class Reference

A [StatusCondition](#) is a condition associated with an [Entity](#). The **trigger_value** is driven by the communication status of the associated [Entity](#).

Inheritance diagram for StatusCondition:



Public Member Functions

- `ReturnCode_t set_enabled_statuses (long mask)`
- `long get_enabled_statuses ()`
- `Entity get_entity ()`

2.18.1 Detailed Description

A [StatusCondition](#) is a condition associated with an [Entity](#). The **trigger_value** is driven by the communication status of the associated [Entity](#).

See also

[com.toc.coredx.DDS.Constants](#) for public Status constants

2.18.2 Member Function Documentation

2.18.2.1 `long get_enabled_statuses ( )`

This routine returns the statuses which are enabled in this [StatusCondition](#). The statuses are returned as a bitmask.

See also

DDS for a list of public Status constants

2.18.2.2 `Entity get_entity ( )`

This routine returns the single entity associated with this [StatusCondition](#).

2.18.2.3 `ReturnCode_t set_enabled_statuses ( long mask )`

This routine sets the statuses which are enabled in this [StatusCondition](#). The statuses are provided as a bitmask.

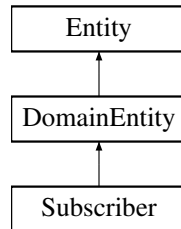
See also

DDS for a list of public Status constants

2.19 Subscriber Class Reference

The [Subscriber](#) configures, creates, manages and destroys DataReaders.

Inheritance diagram for Subscriber:



Public Member Functions

- `ReturnCode_t enable ()`
- `DataReader create_datareader (TopicDescription topic, DataReaderQos qos, DataReaderListener listener, long mask)`
- `ReturnCode_t delete_datareader (DataReader datareader)`
- `ReturnCode_t delete_contained_entities ()`
- `DataReader lookup_datareader (String topic_name)`
- `ReturnCode_t get_datareaders (Vector readers, long sample_states, long view_states, long instance_states)`
- `DomainParticipant get_participant ()`
- `ReturnCode_t set_qos (SubscriberQos qos)`
- `ReturnCode_t get_qos (SubscriberQos qos)`
- `ReturnCode_t set_listener (SubscriberListener new_listener, long mask)`
- `SubscriberListener get_listener ()`
- `ReturnCode_t begin_access ()`
- `ReturnCode_t end_access ()`
- `ReturnCode_t set_default_datareader_qos (DataReaderQos qos)`
- `ReturnCode_t get_default_datareader_qos (DataReaderQos qos)`
- `ReturnCode_t copy_from_topic_qos (DataReaderQos qos, TopicQos topic_qos)`

2.19.1 Detailed Description

The [Subscriber](#) configures, creates, manages and destroys DataReaders.

2.19.2 Member Function Documentation

2.19.2.1 `ReturnCode_t begin_access ( )`

This operation indicates that the application is about to access the data samples in any of the [DataReader](#) objects contained in the [Subscriber](#) `s`. The application is expected to use this operation only if PRESENTATION QosPolicy of the [Subscriber](#) has the `access_scope` set to GROUP. [Otherwise this routine has no effect.]

Not Yet Supported This is currently unsupported in the Java language binding.

2.19.2.2 `ReturnCode_t copy_from_topic_qos ( DataReaderQos qos, TopicQos topic_qos )`

This operation copies the QoS settings in **a_topic_qos** to the corresponding settings in **a_datareader_qos**. The **a_datareader_qos** parameter is populated with a copy of the QoS policies from the **a_topic_qos** structure. QoS entries in the datareader qos structure will be overwritten with the values from the topic.

2.19.2.3 `DataReader create_datareader ( TopicDescription topic, DataReaderQos qos, DataReaderListener listener, long mask )`

This operation creates a [DataReader](#). The created [DataReader](#) is contained within the [Subscriber s](#). It is associated with the [Topic](#), [ContentFilteredTopic](#), or [MultiTopic](#) indicated by **a_topic**, and has the [DataReaderQos](#) indicated by **qos**. The **qos** argument may be passed `DDS.DATAREADER_QOS_DEFAULT`, which indicates that the [Subscriber](#) should use its currently configured default data reader QoS values. The [DataReaderListener a_listener](#), is installed at creation time.

The created [DataReader](#) (if not NULL) must be destroyed by a call to [Subscriber.delete_datareader\(\)](#).

This routine will fail if the provided QoS settings are internally inconsistent. In this case, the routine will return **NULL**.

See also

DDS for a list of STATUS constants available to create the mask

2.19.2.4 `ReturnCode_t delete_contained_entities ( )`

This operation deletes all the DataReaders created by means of the [Subscriber.create_datareader\(\)](#) operation on the [Subscriber s](#). This routine will recursively call the corresponding [delete_contained_entities\(\)](#) operation on each of the contained [DataReader](#) objects. If successful, this operation will recursively delete all objects contained with this [Subscriber](#). After successful execution, the application may delete the [Subscriber](#) by calling [DomainParticipant.delete_subscriber\(\)](#).

If any of the objects cannot be deleted, this routine will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`.

2.19.2.5 `ReturnCode_t delete_datareader ( DataReader datareader )`

This operation deletes a [DataReader](#). If the provided [DataReader a_datareader](#) was not created by [Subscriber s](#), the routine will fail and will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`. If the indicated [DataReader](#) has any outstanding [ReadCondition](#) or [QueryCondition](#) objects the routine will fail and will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`. If the indicated [DataReader](#) has any outstanding 'loans' (from `read()` or `take()` operations), the routine will fail and will return `ReturnCode_t.RETCODE_PRECONDITION_NOT_MET`.

2.19.2.6 `ReturnCode_t enable ( )`

Enables the [Subscriber](#). A [Subscriber](#) is created either enabled or not based on the [DomainParticipantQos](#) setting **entity_factory**. When a [Subscriber](#) is not enabled, only the following sub-set of all [Subscriber](#) operations are legal:

- operations to get and set QoS policies,
- factory operations (create, delete),
- [get_statuscondition\(\)](#),
- [get_status_changes\(\)](#),

- lookup operations

Any other [Subscriber](#) operation may return the `ReturnCode_t.RETCODE_NOT_ENABLED` error. `Subscriber_enable()` may be called on an already enabled [Subscriber](#) [it will have no effect].

2.19.2.7 `ReturnCode_t end_access ( )`

This operation closes a corresponding [Subscriber.begin_access\(\)](#).

Not Yet Supported This is currently unsupported in the Java language binding.

2.19.2.8 `ReturnCode_t get_datareaders ( Vector readers, long sample_states, long view_states, long instance_states )`

This operation allows the application to access [DataReader](#) objects that contain samples with the specified **sample_states**, **view_states**, and **instance_states**.

If the PRESENTATION QoSPolicy of the [Subscriber](#) to which the [DataReader](#) belongs has the `access_scope` set to GROUP, this operation should be invoked only inside a `begin_access/end_access` block. Otherwise it will return the error `PRECONDITION_NOT_MET`.

The returned collection of [DataReader](#) objects may either be a set (containing each [DataReader](#) at most once in no specified order), or a list (containing each [DataReader](#) one or more times in a specific order).

1. If PRESENTATION `access_scope` is INSTANCE or TOPIC, the returned collection is a set.
2. If PRESENTATION `access_scope` is GROUP and `ordered_access` is set to TRUE, then the returned collection is a list.

This difference supports alternate access mechanisms.

Not Yet Supported This is currently unsupported in the Java language binding.

2.19.2.9 `ReturnCode_t get_default_datareader_qos ( DataReaderQos qos )`

Provides access to the default [DataReaderQos](#) settings held in the [Subscriber s](#). The provided **qos** argument is populated with the current default qos settings.

2.19.2.10 `SubscriberListener get_listener ( )`

This operation returns the currently installed [SubscriberListener](#).

2.19.2.11 `DomainParticipant get_participant ( )`

This operation returns the [DomainParticipant](#) this [Subscriber](#) belongs to.

2.19.2.12 `ReturnCode_t get_qos ( SubscriberQos qos )`

Returns the current [SubscriberQos](#) settings held in the [Subscriber s](#). The **qos** parameter is populated with a copy of the current [Subscriber](#) QoS properties.

2.19.2.13 `DataReader` lookup_datareader (*String topic_name*)

This operation retrieves a previously-created `DataReader` contained in the `Subscriber`, attached to a `Topic` named **topic_name**. If multiple `DataReaders` are found, one of them will be returned. If no matching `DataReader` is found, this routine will return **NULL**.

This routine is useful to obtain access to a particular built-in `DataReader`.

2.19.2.14 `ReturnCode_t` set_default_datareader_qos (`DataReaderQos qos`)

Sets the default `DataReaderQos` held in the `Subscriber`. This default qos will be used during subsequent calls to `Subscriber.create_datareader()` if the special `DDS.DATAREADER_QOS_DEFAULT` value is provided for **qos**. This routine may fail if the provided **qos** argument is not internally consistent. In this case, `ReturnCode_t.RETCODE_INCONSISTENT_POLICY` will be returned, and no changes will be made to the `Subscriber`.

2.19.2.15 `ReturnCode_t` set_listener (`SubscriberListener new_listener`, long *mask*)

Installs a `SubscriberListener` on `Subscriber s`. Only one listener may be attached to a `Subscriber` at a time. A call to `set_listener()` will replace any current listener with **a_listener**.

a_listener can be **NULL**, which indicates a listener that does nothing.

See also

DDS for a list of available `STATUS` constants which may be or'ed together to construct the mask.

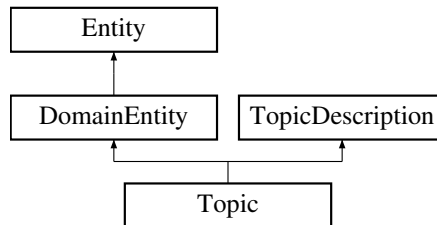
2.19.2.16 `ReturnCode_t` set_qos (`SubscriberQos qos`)

Sets the `SubscriberQos` values. These QoS values affect the behavior of the `Subscriber`. This routine may fail if the provided **qos** argument is not internally consistent. In this case, `ReturnCode_t.RETCODE_INCONSISTENT_POLICY` will be returned, and no changes will be made to the `Subscriber` QoS.

2.20 Topic Class Reference

[Topic](#) is the basic description of data to be published or subscribed. A topic is identified by a **name** and a **type**. A [Topic](#) is created by calling [DomainParticipant.create_topic\(\)](#). Prior to creating a [Topic](#), the associated data type must be registered with the [DomainParticipant](#) via a call to the `TypeSupportXYZ.register_type()` function. [The `register_type()` function is auto-generated 'type-specific' code.].

Inheritance diagram for Topic:



Public Member Functions

- `ReturnCode_t enable ()`
- `DomainParticipant get_participant ()`
This operation returns the parent [DomainParticipant](#) of the [Topic](#).
- `String get_type_name ()`
*This operation returns **type_name** of the [Topic](#).*
- `String get_name ()`
*This operation returns **topic_name** of the [Topic](#).*
- `ReturnCode_t set_qos (TopicQos qos)`
- `ReturnCode_t get_qos (TopicQos qos)`
- `ReturnCode_t set_listener (TopicListener new_listener, long mask)`
- `TopicListener get_listener ()`
- `ReturnCode_t get_inconsistent_topic_status (InconsistentTopicStatus status)`

2.20.1 Detailed Description

[Topic](#) is the basic description of data to be published or subscribed. A topic is identified by a **name** and a **type**. A [Topic](#) is created by calling [DomainParticipant.create_topic\(\)](#). Prior to creating a [Topic](#), the associated data type must be registered with the [DomainParticipant](#) via a call to the `TypeSupportXYZ.register_type()` function. [The `register_type()` function is auto-generated 'type-specific' code.].

2.20.2 Member Function Documentation

2.20.2.1 `ReturnCode_t enable ( )`

Enables the [Topic](#). A [Topic](#) is created either enabled or not based on the [DomainParticipantQos](#) setting **entity_factory**. When a [Topic](#) is not enabled, only the following sub-set of all [Topic](#) operations are legal:

- operations to get and set QoS policies,
- `get_name()`, `get_type_name()`, `get_participant()`

- `get_statuscondition()`,
- `get_status_changes()`,

Any other operation may return the `ReturnCode_t.RETURNCODE_NOT_ENABLED` error. `Topic.enable()` may be called on an already enabled `Topic` [it will have no effect].

2.20.2.2 `ReturnCode_t get_inconsistent_topic_status ( InconsistentTopicStatus status )`

Provides access to the `InconsistentTopicStatus` of the `Topic`. As a side-effect, this routine will reset the `total_count_change` status field to zero.

2.20.2.3 `TopicListener get_listener ( )`

This operation returns the currently installed `TopicListener`.

2.20.2.4 `ReturnCode_t get_qos ( TopicQos qos )`

Returns the current `TopicQos` settings held in the `Topic t`. This routine copies the `Topic` QoS properties into `qos`.

2.20.2.5 `ReturnCode_t set_listener ( TopicListener new_listener, long mask )`

Installs a `TopicListener` on `Topic t`. Only one listener may be attached to a `Topic` at a time. A call to `set_listener()` will replace any current listener with `a_listener`.

`a_listener` can be NULL, which indicates a listener that does nothing.

The infrastructure will make an internal copy of the listener structure so that it need not be persisted by the application.

See also

DDS for a list of available STATUS constants which may be or'ed together to construct the mask.

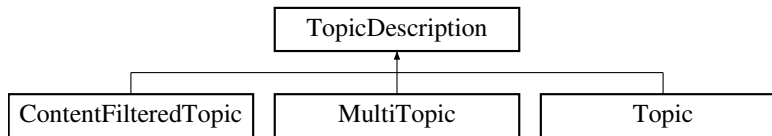
2.20.2.6 `ReturnCode_t set_qos ( TopicQos qos )`

Sets the `TopicQos` values. These QoS values affect the behavior of the `Topic`. This routine may fail if the provided `qos` argument is not internally consistent. In this case, `ReturnCode_t.RETCODE_INCONSISTENT_POLICY` will be returned, and no changes will be made to the `Topic` QoS.

2.21 TopicDescription Interface Reference

[TopicDescription](#) is an abstract that provides the foundation for [Topic](#), [ContentFilteredTopic](#), and [MultiTopic](#).

Inheritance diagram for TopicDescription:



Public Member Functions

- [DomainParticipant](#) [get_participant](#) ()
- String [get_type_name](#) ()
- String [get_name](#) ()

2.21.1 Detailed Description

[TopicDescription](#) is an abstract that provides the foundation for [Topic](#), [ContentFilteredTopic](#), and [MultiTopic](#).

2.21.2 Member Function Documentation

2.21.2.1 String [get_name](#) ()

This operation returns **topic name** of the provided [TopicDescription](#).

Implemented in [Topic](#), [ContentFilteredTopic](#), and [MultiTopic](#).

2.21.2.2 [DomainParticipant](#) [get_participant](#) ()

This operation returns [DomainParticipant](#) that owns the [TopicDescription](#).

Implemented in [Topic](#), [ContentFilteredTopic](#), and [MultiTopic](#).

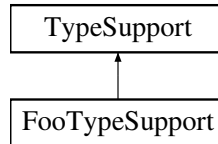
2.21.2.3 String [get_type_name](#) ()

Gets the **type_name** of the provided [TopicDescription](#).

Implemented in [Topic](#), [ContentFilteredTopic](#), and [MultiTopic](#).

2.22 TypeSupport Interface Reference

Inheritance diagram for TypeSupport:



Public Member Functions

- `ReturnCode_t register_type (DomainParticipant dp, String type_name)`
- `String get_typename ()`
- `String get_fully_qualified_typename ()`
- `DDS.DynamicType get_type ()`
- `long getCTypeSupport ()`
- `DataReader create_datareader (Subscriber sub, TopicDescription td, SWIGTYPE_p__DataReader jni_dr)`
- `DataWriter create_datawriter (Publisher pub, Topic topic, SWIGTYPE_p__DataWriter jni_dw)`

2.22.1 Detailed Description

The [TypeSupport](#) interface is implemented by an application defined data type specific [TypeSupport](#) class. This provides a mechanism for an application to register an application defined data type with CoreDX DDS.

See also

[com.toc.coredx.DDS.FooTypeSupport FooTypeSupport](#)

2.22.2 Member Function Documentation

2.22.2.1 DataReader create_datareader (Subscriber sub, TopicDescription td, SWIGTYPE_p__DataReader jni_dr)

Private.

2.22.2.2 DataWriter create_datawriter (Publisher pub, Topic topic, SWIGTYPE_p__DataWriter jni_dw)

Private.

2.22.2.3 String get_fully_qualified_typename ()

Returns the fully-qualified name of the data type supported by this object.

2.22.2.4 DDS.DynamicType get_type ()

Returns a DynamicType instance that represents the data type supported by this [TypeSupport](#) instance.

2.22.2.5 String get_typename ()

Returns the default name of the data type supported by this object.

2.22.2.6 long getCTypeSupport ()

Private.

2.22.2.7 ReturnCode_t register_type (DomainParticipant *dp*, String *type_name*)

Registers a new Application Defined Data Type with the DDS infrastructure. This method is implemented by the application specific [TypeSupport](#) class generated by **coredx_ddl**.

The type is registered under the name 'type_name'. If the 'type_name' parameter is null, then the default name will be used.

See also

[com.toc.coredx.DDS.FooTypeSupport FooTypeSupport](#)

Implemented in [FooTypeSupport](#).

2.23 WaitSet Class Reference

A `DDS_WaitSet` maintains a set of [Condition](#) objects and allows the application to wait until one or more of them have a **trigger_value** of `TRUE`.

Public Member Functions

- [WaitSet](#) ()
- void [destroy](#) ()
- `ReturnCode_t` [attach_condition](#) ([Condition](#) c)
- `ReturnCode_t` [detach_condition](#) ([Condition](#) c)
- `ReturnCode_t` [wait](#) (Vector active_conditions, [Duration_t](#) timeout)
- `ReturnCode_t` [get_conditions](#) (Vector attached_conditions)

2.23.1 Detailed Description

A `DDS_WaitSet` maintains a set of [Condition](#) objects and allows the application to wait until one or more of them have a **trigger_value** of `TRUE`.

Multiple conditions may be attached to a [WaitSet](#).

See also

[Condition](#)

2.23.2 Constructor & Destructor Documentation

2.23.2.1 `WaitSet` ()

Constructor.

2.23.3 Member Function Documentation

2.23.3.1 `ReturnCode_t attach_condition` (`Condition` c)

Adds the condition **c** to the [WaitSet](#). If another thread is currently 'waiting' on the [WaitSet](#), this newly added condition will unblock that thread if its **trigger_value** is `TRUE`.

2.23.3.2 `void destroy` ()

Destructor. Releases internal resources associated with the [WaitSet](#). This [WaitSet](#) is destroyed, and must not be used after this call returns.

2.23.3.3 `ReturnCode_t detach_condition` (`Condition` c)

Adds the condition **c** to the [WaitSet](#). If another thread is currently 'waiting' on the [WaitSet](#), this newly added condition will unblock that thread if its **trigger_value** is `TRUE`.

2.23.3.4 ReturnCode_t get_conditions (Vector *attached_conditions*)

Retrieves the current list of attached conditions. Populates the **attached_conditions** sequence.

2.23.3.5 ReturnCode_t wait (Vector *active_conditions*, Duration_t *timeout*)

Causes the controlling thread to block until an attached condition is triggered or **timeout** elapses.

A return value of DDS_RETCODE_OK indicates that one or more of the attached conditions evaluated to TRUE. Those 'active' conditions are included in the 'out' parameter **active_conditions**.

A return value of DDS_RETCODE_TIMEOUT indicates that the timeout period elapsed without any of the conditions evaluating to TRUE.

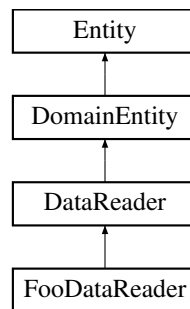
Chapter 3

Example DDS Generated Code

3.1 FooDataReader Class Reference

The [FooDataReader](#) is an example specialized [DataReader](#) (generated by the `coredx_ddl` compiler) for reading data samples of type 'Foo'.

Inheritance diagram for FooDataReader:



Public Member Functions

- `ReturnCode_t` [get_key_value](#) (Foo key_holder, [InstanceHandle_t](#) handle)
- [InstanceHandle_t](#) [lookup_instance](#) (Foo instance_data)
- `ReturnCode_t` [return_loan](#) (FooSeq received_data, SampleInfoSeq sample_infos)
- `ReturnCode_t` [read](#) (FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states)
- `ReturnCode_t` [take](#) (FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states)
- `ReturnCode_t` [read_w_condition](#) (FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, [ReadCondition](#) condition)
- `ReturnCode_t` [take_w_condition](#) (FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, [ReadCondition](#) condition)
- `ReturnCode_t` [read_next_sample](#) (Foo received_data, [SampleInfo](#) sample_info)
- `ReturnCode_t` [take_next_sample](#) (Foo received_data, [SampleInfo](#) sample_info)
- `ReturnCode_t` [read_instance](#) (FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, [InstanceHandle_t](#) handle, long sample_states, long view_states, long instance_states)

- `ReturnCode_t take_instance` (`FooSeq received_data`, `SampleInfoSeq sample_infos`, `int max_samples`, `InstanceHandle_t handle`, `long sample_states`, `long view_states`, `long instance_states`)
- `ReturnCode_t read_next_instance` (`FooSeq received_data`, `SampleInfoSeq sample_infos`, `int max_samples`, `InstanceHandle_t prev_handle`, `long sample_states`, `long view_states`, `long instance_states`)
- `ReturnCode_t take_next_instance` (`FooSeq received_data`, `SampleInfoSeq sample_infos`, `int max_samples`, `InstanceHandle_t prev_handle`, `long sample_states`, `long view_states`, `long instance_states`)
- `ReturnCode_t read_next_instance_w_condition` (`FooSeq received_data`, `SampleInfoSeq sample_infos`, `int max_samples`, `InstanceHandle_t handle`, `ReadCondition condition`)
- `ReturnCode_t take_next_instance_w_condition` (`FooSeq received_data`, `SampleInfoSeq sample_infos`, `int max_samples`, `InstanceHandle_t handle`, `ReadCondition condition`)

3.1.1 Detailed Description

The `FooDataReader` is an example specialized `DataReader` (generated by the `coredx_ddl` compiler) for reading data samples of type 'Foo'.

The `FooDataReader` is an implementation of the base `DataReader` class that has been extended to support the 'Foo' data type.

Note: the `FooDataReader` is included here for documentation purposes only, and does not really exist in the `com.toc.coredx.DDS` package.

3.1.2 Member Function Documentation

3.1.2.1 `ReturnCode_t get_key_value ( Foo key_holder, InstanceHandle_t handle )`

This routine will populate the data structure indicated by `key_holder` with the key information identified by `handle`.

This routine is data type specific. The generated type specific `DataReader` includes an implementation of this routine which should be used to support type-safety.

3.1.2.2 `InstanceHandle_t lookup_instance ( Foo instance_data )`

Returns the handle that identifies the data instance provided in `instance_data`. The 'key' field values of the data are associated with a unique handle.

This routine is data type specific. The generated type specific `DataReader` includes an implementation of this routine which should be used to support type-safety.

3.1.2.3 `ReturnCode_t read ( FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states )`

This operation accesses a collection of data values (with associated `SampleInfo`) within the `DataReader`. This routine, and the related `take()`, provide the interface for an application to access published data. There are several varieties of `read()` and `take()`, to facilitate different access patterns or approaches.

The primary difference between `read()` and `take()` is that `take()` removes all returned data samples from the `DataReader` while `read()` does not. Sequential `read()` calls will return the same data samples each time (if nothing else changes); while sequential `take()` calls will return data samples for only the first call. Subsequent `take()` calls will return an empty collection (if no new data arrives).

The specific behavior of `read()` depends on several things: input parameters, the QoS settings of the `DataReader`, and the state of received data. First, the `received_data` and `sample_infos` arguments affect the following:

- how many samples are returned, and
- whether the returned data should be 'loaned' or copied.

The argument **max_samples** is used to further limit the number of samples returned.

The **sample_states**, **view_states**, and **instance_states** arguments are used to selectively add data samples to the returned collections. These arguments indicate the desired 'states' for data samples and instances. These state arguments are bit masks; they can be the bit-wise OR of several individual state flags or they may use the special 'ANY' constants (e.g.: DDS.ANY_SAMPLE_STATE). Only samples that have a matching state for all three categories are added to the returned collection.

The order of samples in the returned collections is determined by the **PRESENTATION** and **DESTINATION_ORDER** QoS policies.

The returned collection is held in **received_data** and **sample_infos**. These two sequences operate together to represent a sequence of pairs (data, [SampleInfo](#)). Each data item in **received_data** has a corresponding entry in **sample_infos** that provides associated 'meta-data'. See [SampleInfo](#) for a description of this meta-data.

In CoreDX DDS, the returned sequences contain 'loaned' data. This provides zero-copy access to the data, and provides a very efficient data access mechanism. Because the data is 'loaned' to the application, the application is required to indicate when it is finished accessing the data. This is accomplished by calling [return_loan\(\)](#).

The [read\(\)](#) operation will set the [SampleInfo.sample_state](#) to DDS.READ_SAMPLE_STATE.

The [read\(\)](#) operation may set the [SampleInfo.view_state](#) to DDS.NOT_NEW_VIEW_STATE, if a sample of the most recent generation of the instance is read.

If there is no data found, then the [read\(\)](#) will return RETCODE_NO_DATA.

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

3.1.2.4 `ReturnCode_t read_instance ( FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t handle, long sample_states, long view_states, long instance_states )`

This operation accesses a collection of data values (with associated [SampleInfo](#)), belonging to a particular instance, within the [DataReader](#).

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

See also

[com.toc.coredx.DDS.FooDataReader::read\(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states\) read\(\)](#)

3.1.2.5 `ReturnCode_t read_next_instance ( FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t prev_handle, long sample_states, long view_states, long instance_states )`

This operation accesses a collection of data values (with associated [SampleInfo](#)), instance by instance, within the [DataReader](#). To start, pass HANDLE_NIL for parameter prev_handle. After the last instance (in order) has been taken, this routine will return RETCODE_NO_DATA.

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

See also

[com.toc.coredx.DDS.FooDataReader::read](#)(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states) [read\(\)](#)

3.1.2.6 ReturnCode_t read_next_instance_w_condition (FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t handle, ReadCondition condition)

This operation accesses a collection of data values (with associated [SampleInfo](#)), instance by instance subject to a filter, within the [DataReader](#).

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

See also

[com.toc.coredx.DDS.FooDataReader::read](#)(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states) [read\(\)](#)

3.1.2.7 ReturnCode_t read_next_sample (Foo received_data, SampleInfo sample_info)

This operation accesses a data value (with associated [SampleInfo](#)) within the [DataReader](#).

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

See also

[com.toc.coredx.DDS.FooDataReader::read](#)(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states) [read\(\)](#)

3.1.2.8 ReturnCode_t read_w_condition (FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, ReadCondition condition)

This operation accesses a collection of data values (with associated [SampleInfo](#)), subject to a filter, within the [DataReader](#).

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

See also

[com.toc.coredx.DDS.FooDataReader::read](#)(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states) [read\(\)](#)

3.1.2.9 ReturnCode_t return_loan (FooSeq received_data, SampleInfoSeq sample_infos)

Returns data and sample_info values to a [DataReader](#). When an application calls [DataReader read\(\)](#) or [take\(\)](#) operations, these routines 'loan' data and [SampleInfo](#) values to the application. [This is an optimization that avoids extra copies of data.] The application must return this 'loaned' data to the [DataReader](#). The [return_loan\(\)](#) routine indicates to the [DataReader](#) that the application no longer requires access to the data and sample_infos.

A call to [return_loan\(\)](#) operation must be made only if previous [read\(\)](#) or [take\(\)](#) calls 'loaned' data to the application. See [read\(\)](#) for a discussion of when data is 'loaned'.

If the **received_data** or **sample_infos** parameters provided do not identify data obtained from [DataReader dr](#), then the error RETCODE_PRECONDITION_NOT_MET will be returned.

A [DataReader](#) cannot be deleted if any 'loans' are outstanding.

See also

[com.toc.coredx.DDS.FooDataReader::read\(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states\) read\(\)](#)

3.1.2.10 ReturnCode_t take (FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states)

This operation takes a collection of data values (with associated [SampleInfo](#)) from the [DataReader](#). This routine, and the related [read\(\)](#), provide the interface for an application to access published data. There are several varieties of [read\(\)](#) and [take\(\)](#), to facilitate different access patterns or approaches.

The primary difference between [read\(\)](#) and [take\(\)](#) is that [take\(\)](#) removes all returned data samples from the [DataReader](#) while [read\(\)](#) does not. Sequential [read\(\)](#) calls will return the same data samples each time (if nothing else changes); while sequential [take\(\)](#) calls will return data samples for only the first call. Subsequent [take\(\)](#) calls will return an empty collection (if no new data arrives).

The specific behavior of [take\(\)](#) depends on several things: input parameters, the QoS settings of the [DataReader](#), and the state of received data. First, the **received_data** and **sample_infos** arguments affect the following:

- how many samples are returned, and
- whether the returned data should be 'loaned' or copied.

The argument **max_samples** is used to further limit the number of samples returned.

The **sample_states**, **view_states**, and **instance_states** arguments are used to selectively add data samples to the returned collections. These arguments indicate the desired 'states' for data samples and instances. These state arguments are bit masks; they can be the bit-wise OR of several individual state flags or they may use the special 'ANY' constants (e.g.: DDS.ANY_SAMPLE_STATE). Only samples that have a matching state for all three categories are added to the returned collection.

The order of samples in the returned collections is determined by the **PRESENTATION** and **DESTINATION_ORDER** QoS policies.

The returned collection is held in **received_data** and **samples_infos**. These two sequences operate together to represent a sequence of pairs (data, [SampleInfo](#)). Each data item in **received_data** has a corresponding entry in **sample_infos** that provides associated 'meta-data'. See [SampleInfo](#) for a description of this meta-data.

In CoreDX DDS, the returned sequences contain 'loaned' data. This provides zero-copy access to the data, and provides a very efficient data access mechanism. Because the data is 'loaned' to the application, the application is required to indicate when it is finished accessing the data. This is accomplished by calling [return_loan\(\)](#).

The [take\(\)](#) operation will set the [SampleInfo.sample_state](#) to DDS.READ_SAMPLE_STATE.

The [take\(\)](#) operation may set the [SampleInfo.view_state](#) to DDS.NOT_NEW_VIEW_STATE, if a sample of the most recent generation of the instance is taken.

If there is no data found, then the [take\(\)](#) will return RETCODE_NO_DATA.

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

3.1.2.11 `ReturnCode_t take_instance ( FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t handle, long sample_states, long view_states, long instance_states )`

This operation takes a collection of data values (with associated [SampleInfo](#)), belonging to a particular instance, from the [DataReader](#).

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

See also

[com.toc.coredx.DDS.FooDataReader::take](#)(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states) [take\(\)](#)

3.1.2.12 `ReturnCode_t take_next_instance ( FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t prev_handle, long sample_states, long view_states, long instance_states )`

This operation takes a collection of data values (with associated [SampleInfo](#)), instance by instance, from the [DataReader](#). To start, pass HANDLE_NIL for parameter prev_handle. After the last instance (in order) has been taken, this routine will return RETCODE_NO_DATA.

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

See also

[com.toc.coredx.DDS.FooDataReader::take](#)(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states) [take\(\)](#)

3.1.2.13 `ReturnCode_t take_next_instance_w_condition ( FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, InstanceHandle_t handle, ReadCondition condition )`

This operation takes a collection of data values (with associated [SampleInfo](#)), instance by instance subject to a filter, from the [DataReader](#).

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

See also

[com.toc.coredx.DDS.FooDataReader::take](#)(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states) [take\(\)](#)

3.1.2.14 `ReturnCode_t take_next_sample ( Foo received_data, SampleInfo sample_info )`

This operation takes a data value (with associated [SampleInfo](#)) from the [DataReader](#).

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

See also

[com.toc.coredx.DDS.FooDataReader::take](#)(FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, long sample_states, long view_states, long instance_states) [take\(\)](#)

3.1.2.15 `ReturnCode_t take_w_condition ( FooSeq received_data, SampleInfoSeq sample_infos, int max_samples, ReadCondition condition )`

This operation takes a collection of data values (with associated [SampleInfo](#)), subject to a filter, from the [DataReader](#).

This routine is data type specific. The generated type specific [DataReader](#) includes an implementation of this routine which should be used to support type-safety.

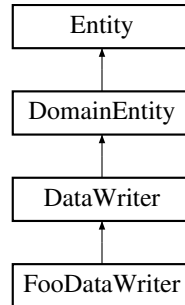
See also

[com.toc.coredx.DDS.FooDataReader::take](#)(FooSeq *received_data*, SampleInfoSeq *sample_infos*, int *max_samples*, long *sample_states*, long *view_states*, long *instance_states*) [take\(\)](#)

3.2 FooDataWriter Class Reference

The [FooDataWriter](#) is an example specialized [DataWriter](#) (generated by the `coredx_ddl` compiler) for writing data samples of type 'Foo'.

Inheritance diagram for [FooDataWriter](#):



Public Member Functions

- `ReturnCode_t` [get_key_value](#) (Foo `key_holder`, [InstanceHandle_t](#) `handle`)
- [InstanceHandle_t](#) [lookup_instance](#) (Foo `instance_data`)
- [InstanceHandle_t](#) [register_instance](#) (Foo `instance_data`)
- [InstanceHandle_t](#) [register_instance_w_timestamp](#) (Foo `instance_data`, [Time_t](#) `ts`)
- `ReturnCode_t` [unregister_instance](#) (Foo `instance_data`, [InstanceHandle_t](#) `handle`)
- `ReturnCode_t` [unregister_instance_w_timestamp](#) (Foo `instance_data`, [InstanceHandle_t](#) `handle`, [Time_t](#) `timestamp`)
- `ReturnCode_t` [write](#) (Foo `instance_data`, [InstanceHandle_t](#) `handle`)
- `ReturnCode_t` [write_w_timestamp](#) (Foo `instance_data`, [InstanceHandle_t](#) `handle`, [Time_t](#) `timestamp`)
- `ReturnCode_t` [dispose](#) (Foo `instance_data`, [InstanceHandle_t](#) `handle`)
- `ReturnCode_t` [dispose_w_timestamp](#) (Foo `instance_data`, [InstanceHandle_t](#) `handle`, [Time_t](#) `timestamp`)

3.2.1 Detailed Description

The [FooDataWriter](#) is an example specialized [DataWriter](#) (generated by the `coredx_ddl` compiler) for writing data samples of type 'Foo'.

The [FooDataWriter](#) is an implementation of the base [DataWriter](#) class that has been extended to support the 'Foo' data type.

Note: the [FooDataWriter](#) is included here for documentation purposes only, and does not really exist in the `com.toc.coredx.DDS` package.

3.2.2 Member Function Documentation

3.2.2.1 `ReturnCode_t` [dispose](#) (Foo *instance_data*, [InstanceHandle_t](#) *handle*)

Indicates that the instance no longer exists. If **handle** is not `HANDLE_NIL`, then **handle** must identify a valid instance that has been previously registered or written by this [DataWriter](#). The current time is used as the source timestamp.

3.2.2.2 ReturnCode_t dispose_w_timestamp (Foo instance_data, InstanceHandle_t handle, Time_t timestamp)

Indicates that the instance no longer exists. If **handle** is not HANDLE_NIL, then **handle** must identify a valid instance that has been previously registered or written by this [DataWriter](#). The **source_timestamp** is used as the source timestamp for the published message.

3.2.2.3 ReturnCode_t get_key_value (Foo key_holder, InstanceHandle_t handle)

This routine will populate the data structure indicated by **key_holder** with the key information identified by **handle**.

This routine is data type specific. The generated type specific [DataWriter](#) includes an implementation of this routine which should be used to support type-safety.

3.2.2.4 InstanceHandle_t lookup_instance (Foo instance_data)

Returns the handle that identifies the data instance provided in **instance_data**. The 'key' field values of the data are associated with a unique handle.

3.2.2.5 InstanceHandle_t register_instance (Foo instance_data)

Declares the existence of an instance identified by the 'key fields' in **instance_data**. The handle that uniquely identifies the instance is returned. The current time is used as the source timestamp.

3.2.2.6 InstanceHandle_t register_instance_w_timestamp (Foo instance_data, Time_t ts)

Declares the existence of an instance identified by the 'key fields' in **instance_data**. The handle that uniquely identifies the instance is returned. The **source_timestamp** is used as the source timestamp.

3.2.2.7 ReturnCode_t unregister_instance (Foo instance_data, InstanceHandle_t handle)

Indicates that the writer will no longer be providing updates the specified instance. If **handle** is not HANDLE_NIL, then **handle** must identify a valid instance that has been previously registered or written by this [DataWriter](#). The current time is used as the source timestamp.

3.2.2.8 ReturnCode_t unregister_instance_w_timestamp (Foo instance_data, InstanceHandle_t handle, Time_t timestamp)

Indicates that the writer will no longer be providing updates the specified instance. If **handle** is not HANDLE_NIL, then **handle** must identify a valid instance that has been previously registered or written by this [DataWriter](#). The provided **source_timestamp** is used as the source timestamp.

3.2.2.9 ReturnCode_t write (Foo instance_data, InstanceHandle_t handle)

Publishes the provided **instance_data**. If **handle** is HANDLE_NIL, then the handle is determined from the 'key fields' of **instance_data**.

The current time is used as the source timestamp for the published data.

This routine may block if RELIABILITY kind == RELIABLE, RESOURCE_LIMITS are not UNLIMITED, and the local resources are exhausted. The routine will block at most 'max_blocking_time' as configured in the RELIABILITY QoS. If the routine is still unable to handle the data, after blocking, then RETCODE_TIMEOUT is returned.

The routine may return `RETCODE_OUT_OF_RESOURCES` if it is determined that no amount of blocking will allow the data to be processed.

On success, `RETCODE_OK` is returned.

3.2.2.10 `ReturnCode_t write_w_timestamp ( Foo instance_data, InstanceHandle_t handle, Time_t timestamp )`

Publishes the provided **instance_data**. If **handle** is `HANDLE_NIL`, then the handle is determined from the 'key fields' of **instance_data**.

The **source_timestamp** is used as the source timestamp for the published data.

This routine may block if `RELIABILITY kind == RELIABLE`, `RESOURCE_LIMITS` are not `UNLIMITED`, and the local resources are exhausted. The routine will block at most 'max_blocking_time' as configured in the `RELIABILITY QoS`. If the routine is still unable to handle the data, after blocking, then `RETCODE_TIMEOUT` is returned.

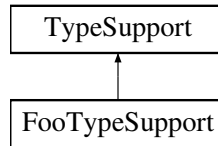
The routine may return `RETCODE_OUT_OF_RESOURCES` if it is determined that no amount of blocking will allow the data to be processed.

On success, `RETCODE_OK` is returned.

3.3 FooTypeSupport Class Reference

The [FooTypeSupport](#) class implements the [TypeSupport](#) interface for the Data Type 'Foo'.

Inheritance diagram for FooTypeSupport:



Public Member Functions

- `ReturnCode_t register_type (DomainParticipant dp, String type_name)`
- `String get_type_name ()`

3.3.1 Detailed Description

The [FooTypeSupport](#) class implements the [TypeSupport](#) interface for the Data Type 'Foo'.

This provides a mechanism for an application to register the 'Foo' application defined data type with CoreDX DDS.

Note: [FooTypeSupport](#) is provided here for illustration purposes and is not really included in the `com.toc.coredx.DDS` package.

See also

[com.toc.coredx.DDS.FooTypeSupport FooTypeSupport](#)

3.3.2 Member Function Documentation

3.3.2.1 String get_type_name ()

Returns the default name of the Data Type supported by this instance of [TypeSupport](#) (in this case, 'Foo').

Implements [TypeSupport](#).

3.3.2.2 ReturnCode_t register_type (DomainParticipant dp, String type_name)

Registers the 'Foo' Data Type with the DDS infrastructure. The type is registered under the name 'type_name'. If the 'type_name' parameter is null, then the default name ('Foo' in this case), will be used.

Implements [TypeSupport](#).

Chapter 4

Entity QoS Collections

4.1 DataReaderQos Class Reference

Structure that holds [DataReader](#) Quality of Service policies.

Public Attributes

- [DurabilityQosPolicy](#) durability
- [DeadlineQosPolicy](#) deadline
- [LatencyBudgetQosPolicy](#) latency_budget
- [LivelinessQosPolicy](#) liveliness
- [ReliabilityQosPolicy](#) reliability
- [DestinationOrderQosPolicy](#) destination_order
- [HistoryQosPolicy](#) history
- [ResourceLimitsQosPolicy](#) resource_limits
- [UserDataQosPolicy](#) user_data
- [OwnershipQosPolicy](#) ownership
- [TimeBasedFilterQosPolicy](#) time_based_filter
- [ReaderDataLifecycleQosPolicy](#) reader_data_lifecycle
- [DataRepresentationQosPolicy](#) representation
- [TypeConsistencyEnforcementQosPolicy](#) type_consistency
- [RpcQosPolicy](#) rpc
- [EntityNameQosPolicy](#) entity_name
- [LoggingQosPolicy](#) logging
- [RTPSReaderQosPolicy](#) rtps_reader

Static Public Attributes

- static final [DataReaderQos](#) DATAREADER_QOS_DEFAULT = null

4.1.1 Detailed Description

Structure that holds [DataReader](#) Quality of Service policies.

See also

```
DataReader::set_qos(DataReaderQos) set_qos()  
DataReader::get_qos(DataReaderQos) get_qos()  
Subscriber::create_datareader(TopicDescription topic, DataReaderQos qos, DataReaderListener listener, long  
mask) create_datareader()  
Subscriber::set_default_datareader_qos(DataReaderQos) set_default_datareader_qos()  
Subscriber::get_default_datareader_qos(DataReaderQos) get_default_datareader_qos()
```

4.1.2 Member Data Documentation

4.1.2.1 DeadlineQosPolicy deadline

The requested update frequency for data instances.

4.1.2.2 DestinationOrderQosPolicy destination_order

The destination order logic requested by the [DataReader](#).

4.1.2.3 DurabilityQosPolicy durability

The durability policy requested by the [DataReader](#).

4.1.2.4 EntityNameQosPolicy entity_name

Controls the [Entity](#) name of the [DataReader](#)

4.1.2.5 HistoryQosPolicy history

The data history requested by the [DataReader](#).

4.1.2.6 LatencyBudgetQosPolicy latency_budget

The latency requested by the [DataReader](#).

4.1.2.7 LivelinessQosPolicy liveliness

The liveliness mechanism requested by the [DataReader](#).

4.1.2.8 LoggingQosPolicy logging

Controls the logging behavior of the [DataReader](#).

4.1.2.9 OwnershipQosPolicy ownership

The type of 'ownership' offered by the [DataReader](#).

4.1.2.10 ReaderDataLifecycleQosPolicy reader_data_lifecycle

Controls the auto-purge behavior of the [DataReader](#).

4.1.2.11 ReliabilityQosPolicy reliability

The transport reliability requested by the [DataReader](#).

4.1.2.12 DataRepresentationQosPolicy representation

Informs DataWriter(s) of the data representation(s) supported by this Reader. This must be consistent with the [Type-Support](#) associated with the Reader's [Topic](#).

4.1.2.13 ResourceLimitsQosPolicy resource_limits

The resource limits set on the [DataReader](#).

4.1.2.14 RpcQosPolicy rpc

Configure RPC relevant settings: instance_name, related_entity, and topic_aliases.

4.1.2.15 RTPSReaderQosPolicy rtps_reader

Controls aspects of the RTPS Reader protocol for the [DataReader](#).

4.1.2.16 TimeBasedFilterQosPolicy time_based_filter

The maximum update frequency required/desired by the [DataReader](#).

4.1.2.17 TypeConsistencyEnforcementQosPolicy type_consistency

Influences the algorithm that matches DataReaders and DataWriters based on their data type compatibility.

4.1.2.18 UserDataQosPolicy user_data

A sequence of octets associated with the [DataReader](#).

4.2 DataWriterQos Class Reference

Structure that holds [DataWriter](#) Quality of Service policies.

Public Attributes

- [DurabilityQosPolicy](#) durability
- [DurabilityServiceQosPolicy](#) durability_service
- [DeadlineQosPolicy](#) deadline
- [LatencyBudgetQosPolicy](#) latency_budget
- [LivelinessQosPolicy](#) liveliness
- [ReliabilityQosPolicy](#) reliability
- [DestinationOrderQosPolicy](#) destination_order
- [HistoryQosPolicy](#) history
- [ResourceLimitsQosPolicy](#) resource_limits
- [TransportPriorityQosPolicy](#) transport_priority
- [LifespanQosPolicy](#) lifespan
- [UserDataQosPolicy](#) user_data
- [OwnershipQosPolicy](#) ownership
- [OwnershipStrengthQosPolicy](#) ownership_strength
- [WriterDataLifecycleQosPolicy](#) writer_data_lifecycle
- [DataRepresentationQosPolicy](#) representation
- [RpcQosPolicy](#) rpc
- [EntityNameQosPolicy](#) entity_name
- [LoggingQosPolicy](#) logging
- [RTPSWriterQosPolicy](#) rtps_writer

Static Public Attributes

- static final [DataWriterQos](#) DATAWRITER_QOS_DEFAULT = null

4.2.1 Detailed Description

Structure that holds [DataWriter](#) Quality of Service policies.

See also

```

DataWriter::set\_qos\(DataWriterQos\) set_qos()
DataWriter::get\_qos\(DataWriterQos\) get_qos()
Publisher::create\_datawriter\(Topic topic, DataWriterQos qos, DataWriterListener listener, long mask\) create_datawriter()
Publisher::set\_default\_datawriter\_qos\(DataWriterQos\) set_default_datawriter_qos()
Publisher::get\_default\_datawriter\_qos\(DataWriterQos\) get_default_datawriter_qos()

```

4.2.2 Member Data Documentation

4.2.2.1 DeadlineQosPolicy deadline

The deadline committed to by the [DataWriter](#).

4.2.2.2 DestinationOrderQosPolicy destination_order

The destination order logic offered by the [DataWriter](#).

4.2.2.3 DurabilityQosPolicy durability

The durability policy offered by the [DataWriter](#).

4.2.2.4 DurabilityServiceQosPolicy durability_service

The durability service configuration offered by the [DataWriter](#).

4.2.2.5 EntityNameQosPolicy entity_name

Controls the [Entity](#) name of the [DataWriter](#)

4.2.2.6 HistoryQosPolicy history

The data history maintained by the [DataWriter](#).

4.2.2.7 LatencyBudgetQosPolicy latency_budget

The latency allowed by the [DataWriter](#).

4.2.2.8 LifespanQosPolicy lifespan

The expiration time for old samples managed by the [DataWriter](#).

4.2.2.9 LivelinessQosPolicy liveliness

The liveliness mechanism offered by the [DataWriter](#).

4.2.2.10 LoggingQosPolicy logging

Controls the logging behavior of the [DataWriter](#).

4.2.2.11 OwnershipQosPolicy ownership

The type of 'ownership' offered by the [DataWriter](#).

4.2.2.12 OwnershipStrengthQosPolicy ownership_strength

The measure of 'ownership strength' offered by the [DataWriter](#).

4.2.2.13 ReliabilityQosPolicy reliability

The transport reliability offered by the [DataWriter](#).

4.2.2.14 DataRepresentationQosPolicy representation

Informs DataReader(s) of the data representation supported by this Writer. This must be consistent with the [TypeSupport](#) associated with the Writer's [Topic](#).

4.2.2.15 ResourceLimitsQosPolicy resource_limits

The resource limits set on the [DataWriter](#).

4.2.2.16 RpcQosPolicy rpc

Configure RPC relevant settings: instance_name, related_entity, and topic_aliases.

4.2.2.17 RTPSWriterQosPolicy rtps_writer

Controls aspects of the RTPS Writer protocol for the [DataWriter](#).

4.2.2.18 TransportPriorityQosPolicy transport_priority

The transport priority supported by the [DataWriter](#).

4.2.2.19 UserDataQosPolicy user_data

A sequence of octets associated with the [DataWriter](#).

4.2.2.20 WriterDataLifecycleQosPolicy writer_data_lifecycle

Indicates if unregistered instances should be automatically disposed by the [DataWriter](#).

4.3 DomainParticipantFactoryQos Class Reference

Structure that holds [DomainParticipantFactory](#) Quality of Service policies.

Public Attributes

- [EntityFactoryQosPolicy](#) entity_factory

4.3.1 Detailed Description

Structure that holds [DomainParticipantFactory](#) Quality of Service policies.

See also

[DomainParticipantFactory::set_qos\(DomainParticipantFactoryQos\)](#) set_qos()
[DomainParticipantFactory::get_qos\(DomainParticipantFactoryQos\)](#) get_qos()

4.3.2 Member Data Documentation

4.3.2.1 EntityFactoryQosPolicy entity_factory

Controls the behavior of the [DomainParticipant](#) 'create' operations.

4.4 DomainParticipantQos Class Reference

Structure that holds [DomainParticipant](#) Quality of Service policies.

Public Member Functions

- [DomainParticipantQos](#) ()

Constructor.

Public Attributes

- [UserDataQosPolicy](#) user_data
- [EntityFactoryQosPolicy](#) entity_factory
- [PropertyQosPolicy](#) properties
- [EntityNameQosPolicy](#) entity_name
- [LoggingQosPolicy](#) logging
- [PeerParticipantQosPolicy](#) peer_participants
- [DiscoveryQosPolicy](#) discovery
- [ThreadModelQosPolicy](#) thread_model

Static Public Attributes

- static final [DomainParticipantQos](#) PARTICIPANT_QOS_DEFAULT = null

4.4.1 Detailed Description

Structure that holds [DomainParticipant](#) Quality of Service policies.

See also

```
DomainParticipant::set_qos(DomainParticipantQos) set_qos()
DomainParticipant::get_qos(DomainParticipantQos) get_qos()
DomainParticipantFactory::create_participant(long domain_id, DomainParticipantQos qos, DomainParticipantListener listener, long mask) create_participant()
DomainParticipantFactory::set_default_participant_qos(DomainParticipantQos) set_default_participant_qos()
DomainParticipantFactory::get_default_participant_qos(DomainParticipantQos) get_default_participant_qos()
```

4.4.2 Member Data Documentation

4.4.2.1 DiscoveryQosPolicy discovery

Controls aspects of the Discovery and Builtin entities

4.4.2.2 EntityFactoryQosPolicy entity_factory

Controls the behavior of the [DomainParticipant](#) **create** operations.

4.4.2.3 EntityNameQosPolicy entity_name

Controls the [Entity](#) name of the [DomainParticipant](#)

4.4.2.4 LoggingQosPolicy logging

Controls the logging behavior of the [DomainParticipant](#)

4.4.2.5 PeerParticipantQosPolicy peer_participants

Specifies a list of [DomainParticipant](#) peers to attempt communication with. If empty, default Discovery is used.

4.4.2.6 PropertyQosPolicy properties

Additional name:value pair properties (if propagate=true, included in discovery)

4.4.2.7 ThreadModelQosPolicy thread_model

Controls the threading behavior of the [DomainParticipant](#)

4.4.2.8 UserDataQosPolicy user_data

A sequence of octets associated with a [DomainParticipant](#).

4.5 PublisherQos Class Reference

Structure that holds [Publisher](#) Quality of Service policies.

Public Attributes

- [PresentationQosPolicy](#) presentation
- [PartitionQosPolicy](#) partition
- [GroupDataQosPolicy](#) group_data
- [EntityFactoryQosPolicy](#) entity_factory
- [EntityNameQosPolicy](#) entity_name
- [LoggingQosPolicy](#) logging

Static Public Attributes

- static final [PublisherQos](#) PUBLISHER_QOS_DEFAULT = null

4.5.1 Detailed Description

Structure that holds [Publisher](#) Quality of Service policies.

See also

```
Publisher::set\_qos\(PublisherQos\) set_qos()  
Publisher::get\_qos\(PublisherQos\) get_qos()  
DomainParticipant::create\_publisher\(PublisherQos qos, PublisherListener listener, long mask\) create_publisher()  
DomainParticipant::set\_default\_publisher\_qos\(PublisherQos\) set_default_publisher_qos()  
DomainParticipant::get\_default\_publisher\_qos\(PublisherQos\) get_default_publisher_qos()
```

4.5.2 Member Data Documentation

4.5.2.1 [EntityFactoryQosPolicy](#) entity_factory

Controls the behavior of the [Publisher.create_datawriter\(\)](#) operation.

4.5.2.2 [EntityNameQosPolicy](#) entity_name

Controls the [Entity](#) name of the [Publisher](#)

4.5.2.3 [GroupDataQosPolicy](#) group_data

A sequence of octets associated with the [Publisher](#).

4.5.2.4 [LoggingQosPolicy](#) logging

Controls the logging behavior of the [Publisher](#)

4.5.2.5 PartitionQosPolicy partition

Establishes a logical data partition. DataWriters and DataReaders that are 'in' the same partition (ie, the partition of the containing [Publisher](#) and [Subscriber](#) match) can communicate. If the partitions do not match, then they cannot communicate.

4.5.2.6 PresentationQosPolicy presentation

Controls the presentation of groups of changes.

See also

[Publisher::begin_coherent_changes\(\)](#) [begin_coherent_changes\(\)](#)
[Publisher::end_coherent_changes\(\)](#) [end_coherent_changes\(\)](#)
[Subscriber::begin_access\(\)](#) [begin_access\(\)](#)
[Subscriber::end_access\(\)](#) [end_access\(\)](#)

4.6 SubscriberQos Class Reference

Structure that holds DDS_Subscriber Quality of Service policies.

Public Attributes

- [PresentationQosPolicy](#) presentation
- [PartitionQosPolicy](#) partition
- [GroupDataQosPolicy](#) group_data
- [EntityFactoryQosPolicy](#) entity_factory
- [EntityNameQosPolicy](#) entity_name
- [LoggingQosPolicy](#) logging

Static Public Attributes

- static final [SubscriberQos](#) SUBSCRIBER_QOS_DEFAULT = null

4.6.1 Detailed Description

Structure that holds DDS_Subscriber Quality of Service policies.

See also

```
Subscriber::set_qos(SubscriberQos)
Subscriber::get_qos(SubscriberQos)
DomainParticipant::create_subscriber(SubscriberQos qos, SubscriberListener listener, long mask)
DomainParticipant::set_default_subscriber_qos(SubscriberQos)
DomainParticipant::get_default_subscriber_qos(SubscriberQos)
```

4.6.2 Member Data Documentation

4.6.2.1 EntityFactoryQosPolicy entity_factory

Controls the behavior of the [Subscriber.create_datareader\(\)](#) operation.

4.6.2.2 EntityNameQosPolicy entity_name

Controls the [Entity](#) name of the [Subscriber](#)

4.6.2.3 GroupDataQosPolicy group_data

A sequence of octets associated with the [Publisher](#).

4.6.2.4 LoggingQosPolicy logging

Controls the logging behavior of the [Subscriber](#)

4.6.2.5 PartitionQosPolicy partition

Establishes a logical data partition. DataWriters and DataReaders that are 'in' the same partition (ie, the partition of the containing [Publisher](#) and [Subscriber](#) match) can communicate. If the partitions do not match, then they cannot communicate.

4.6.2.6 PresentationQosPolicy presentation

Controls the presentation of groups of changes.

See also

[Publisher::begin_coherent_changes\(\)](#)
[Publisher::end_coherent_changes\(\)](#)
[Subscriber::begin_access\(\)](#)
[Subscriber::end_access\(\)](#)

4.7 TopicQos Class Reference

Structure that holds DDS_Topic Quality of Service policies.

Public Attributes

- [TopicDataQosPolicy](#) topic_data
- [DurabilityQosPolicy](#) durability
- [DurabilityServiceQosPolicy](#) durability_service
- [DeadlineQosPolicy](#) deadline
- [LatencyBudgetQosPolicy](#) latency_budget
- [LivelinessQosPolicy](#) liveliness
- [ReliabilityQosPolicy](#) reliability
- [DestinationOrderQosPolicy](#) destination_order
- [HistoryQosPolicy](#) history
- [ResourceLimitsQosPolicy](#) resource_limits
- [TransportPriorityQosPolicy](#) transport_priority
- [LifespanQosPolicy](#) lifespan
- [OwnershipQosPolicy](#) ownership
- [EntityNameQosPolicy](#) entity_name
- [LoggingQosPolicy](#) logging

Static Public Attributes

- static final [TopicQos](#) TOPIC_QOS_DEFAULT = null

4.7.1 Detailed Description

Structure that holds DDS_Topic Quality of Service policies.

See also

```

Topic::set\_qos\(TopicQos\) set_qos()
Topic::get\_qos\(TopicQos\) get_qos()
DomainParticipant::create\_topic(java.lang.String topic_name, java.lang.String type_name, TopicQos DDS Quality
of Service, TopicListener listener, long mask) create_topic()
DomainParticipant::set\_default\_topic\_qos\(TopicQos\) set_default_topic_qos()

```

4.7.2 Member Data Documentation

4.7.2.1 DeadlineQosPolicy deadline

Defines the expected update frequency for data instances within the [Topic](#).

4.7.2.2 DestinationOrderQosPolicy destination_order

The ordering of received samples on the [Topic](#) will be either by SOURCE or RECEPTION timestamp.

4.7.2.3 DurabilityQosPolicy durability

The durability policy of the [Topic](#).

4.7.2.4 DurabilityServiceQosPolicy durability_service

Configures the service supporting the TRANSIENT and PERSISTENT durability kinds.

4.7.2.5 EntityNameQosPolicy entity_name

Controls the [Entity](#) name of the [Topic](#)

4.7.2.6 HistoryQosPolicy history

The amount of historical data maintained for the [Topic](#).

4.7.2.7 LatencyBudgetQosPolicy latency_budget

Identifies the 'urgency' of the data on the [Topic](#). The middleware uses this to batch data samples is possible.

4.7.2.8 LifespanQosPolicy lifespan

The 'expiration time' for old data samples on the [Topic](#).

4.7.2.9 LivelinessQosPolicy liveliness

Identifies the mechanism used to detect and maintain liveliness of DataWriters on the [Topic](#).

4.7.2.10 LoggingQosPolicy logging

Controls the logging behavior of the [Topic](#)

4.7.2.11 OwnershipQosPolicy ownership

The type of 'ownership' expected for data instances on the [Topic](#).

4.7.2.12 ReliabilityQosPolicy reliability

Indicates the level of transport reliability on the [Topic](#).

4.7.2.13 ResourceLimitsQosPolicy resource_limits

The resource limitations for the [Topic](#).

4.7.2.14 TopicDataQosPolicy topic_data

A sequence of octets associated with a [Topic](#).

4.7.2.15 TransportPriorityQosPolicy transport_priority

The priority to be used for messages on the [Topic](#).

Chapter 5

QoS Policies

5.1 DeadlineQosPolicy Class Reference

This QoS policy establishes a minimum update period for data instances.

5.1.1 Detailed Description

This QoS policy establishes a minimum update period for data instances.

DataWriters specify that each data instance will be updated at least once every 'period'. A [DataReader](#) requests that the Writer commit to updating each instance at least once every 'period'.

The policy is compatible (between a Writer and a Reader) if the offered deadline period $\leq$ requested deadline period.

If a [DataWriter](#) fails to meet the update period, then the [DataWriter](#) is notified by a call to the `offered_deadline_missed` listener, and the [DataReader](#) is notified by a call to the `requested_deadline_missed` listener.

See also

```
DataReaderListener::on\_requested\_deadline\_missed\(DataReader the\_reader, RequestedDeadlineMissedStatus status\) on_requested_deadline_missed  
DataWriterListener::on\_offered\_deadline\_missed\(DataWriter writer, OfferedDeadlineMissedStatus status\)  
on_offered_deadline_missed
```

5.2 DataRepresentationQosPolicy Class Reference

This QoS policy establishes the DataRepresentation (data encoding) offered by a [DataWriter](#) or requested by a [DataReader](#).

Public Attributes

- Vector< Short > [value](#)

5.2.1 Detailed Description

This QoS policy establishes the DataRepresentation (data encoding) offered by a [DataWriter](#) or requested by a [DataReader](#).

DataWriters offer a specific single encoding. DataReaders request zero or more encodings.

The policy is compatible between a Writer and a Reader if there is a matching encoding between offered and requested.

5.2.2 Member Data Documentation

5.2.2.1 Vector<Short> value

a sequence of representation values

5.3 DestinationOrderQosPolicy Class Reference

This QoS policy controls how each [Subscriber](#) orders received data samples.

5.3.1 Detailed Description

This QoS policy controls how each [Subscriber](#) orders received data samples.

Can be **BY_RECEPTION_TIMESTAMP** or **BY_SOURCE_TIMESTAMP**. The samples are ordered either based on their order of reception, or by the order of timestamps generated by the source ([DataWriter](#)).

CoreDX DDS currently implements only BY_RECEPTION_TIMESTAMP.

5.4 DiscoveryQosPolicy Class Reference

QoS Policy for configuring aspects of the Discovery and Builtin entities.

5.4.1 Detailed Description

QoS Policy for configuring aspects of the Discovery and Builtin entities.

5.5 DurabilityQosPolicy Class Reference

5.5.1 Detailed Description

The DurabilityQosPolicy controls the durability of data. The DDS API identifies several degrees of data durability.

1. **VOLATILE**: The data is volatile, and is not persisted beyond the initial publication.
2. **TRANSIENT_LOCAL**: The data is persisted locally within the source [DataWriter](#). If that [DataWriter](#) is destroyed, or the containing application exits, the data is no longer available.
3. **TRANSIENT**: The data is persisted beyond the lifecycle of the originating [DataWriter](#), and is available even after that [DataWriter](#) has been destroyed. However, data does not persist a reboot of the computer.
4. **PERSISTENT**: The data is persisted to 'off-line' storage, and is available even after a system reboot.

A [DataWriter](#) can offer to provide a level of durability for data that it generates, and a [DataReader](#) requests the level of durability that it requires. If the [DataReader](#) requests a 'higher' level of durability than that offered by the [DataWriter](#), then the QoS policy is incompatible.

CoreDX DDS currently supports VOLATILE and TRANSIENT_LOCAL.

5.6 DurabilityServiceQosPolicy Class Reference

5.6.1 Detailed Description

The DurabilityServiceQosPolicy supports the durability of data. The DDS API identifies several degrees of data durability.

1. **VOLATILE**: The data is volatile, and is not persisted beyond the initial publication.
2. **TRANSIENT_LOCAL**: The data is persisted locally within the source [DataWriter](#). If that [DataWriter](#) is destroyed, or the containing application exits, the data is no longer available.
3. **TRANSIENT**: The data is persisted beyond the lifecycle of the originating [DataWriter](#), and is available even after that [DataWriter](#) has been destroyed. However, data does not persist a reboot of the computer.
4. **PERSISTENT**: The data is persisted to 'off-line' storage, and is available even after a system reboot.

If a durability of TRANSIENT or PERSISTENT is specified, then a service beyond the source [DataWriter](#) must be established to provide advanced data durability. This QoS policy helps to configure that durability service.

CoreDX DDS currently supports VOLATILE and TRANSIENT_LOCAL.

5.7 EntityFactoryQosPolicy Class Reference

5.7.1 Detailed Description

Controls the behavior of entity factories. Several DDS entities act as factories to create, control, and destroy other DDS entities. For example, the [DomainParticipant](#) is a factory for [Publisher](#), [Subscriber](#), and [Topic](#) entities. This QoS controls the behavior of a factory.

A DDS entity can be enabled at creation time. If the `autoenable_created_entities` is set to **true**, then the factory will automatically call `enable()` on the entity during creation. Otherwise, the entity is created but not enabled. The application must specifically call `enable()` on the returned entity.

5.8 EntityNameQosPolicy Class Reference

5.8.1 Detailed Description

Assigns a name to an [Entity](#). This is a CoreDX DDS extension, and is provided for convenience. [Entity](#) names assigned through this QoS are not used internally by the middleware, they are provided for debug and analysis purposes.

5.9 GroupDataQosPolicy Class Reference

Allows the application to attach arbitrary information to a [Publisher](#) or [Subscriber](#).

5.9.1 Detailed Description

Allows the application to attach arbitrary information to a [Publisher](#) or [Subscriber](#).

The value of the GROUP_DATA is propagated and made available to applications through the built-in topics.

The group data is implemented as a vector of Bytes, and can be used to store any application defined data.

This QoS can be used by an application in combination with the [DataReaderListener](#) and [DataWriterListener](#) to implement conditional matching policies.

5.10 HistoryQosPolicy Class Reference

Controls the ammount of historical data maintained by a [DataReader](#) or [DataWriter](#).

5.10.1 Detailed Description

Controls the ammount of historical data maintained by a [DataReader](#) or [DataWriter](#).

The history kind can be set to KEEP_ALL_HISTORY_QOS or KEEP_LAST_HISTORY_QOS.

If the history kind is KEEP_LAST, then the history depth field is used to determine the number of samples to keep. The default setting is KEEP_LAST with a depth of 1. This means that for a single sample (the most recent) is kept for each data instance.

If the history kind is KEEP_ALL, then the depth field is unused, and all samples will be kept, within the limits set by the ResourceLimits QoS policy.

This QoS policy must be consistent with the ResourceLimits policy.

See also

[ResourceLimitsQosPolicy](#)

5.11 LatencyBudgetQosPolicy Class Reference

Specifies allowable latency.

5.11.1 Detailed Description

Specifies allowable latency.

The latency budget is used internally by CoreDX DDS to optimize data messaging. If the middleware is provided a latency budget that is non-zero, then several internal operations can be optimized to reduce message and processing overhead.

To be compatible, the [DataWriter](#) must offer a `latency_budget` that is equal to or smaller than that requested by the [DataReader](#).

5.12 LifespanQosPolicy Class Reference

Specifies the maximum duration of validity of the data written by the [DataWriter](#).

5.12.1 Detailed Description

Specifies the maximum duration of validity of the data written by the [DataWriter](#).

The default value of the lifespan duration is infinite.

5.13 LivelinessQosPolicy Class Reference

Determines the mechanism and parameters used by the application to determine whether an [Entity](#) is alive.

5.13.1 Detailed Description

Determines the mechanism and parameters used by the application to determine whether an [Entity](#) is alive.

The QoS defines the kind of liveliness (how liveliness is maintained) as well as a 'lease_duration' which specifies an interval for checking liveliness. The liveliness kind can be one of the following:

- AUTOMATIC
- MANUAL_BY_PARTICIPANT
- MANUAL_BY_TOPIC

The lease_duration indicates the maximum interval between liveliness indications from the publishing entity. If this period elapses with no indication from the publishing entity, then the entity is considered not alive.

Changes in liveliness are indicated to the application through status changes and the [DataWriterListener.on_liveliness_lost](#) and [DataReaderListener.on_liveliness_changed](#) listeners.

See also

[DataReaderListener::on_liveliness_changed](#)([DataReader](#) the_reader, [LivelinessChangedStatus](#) DDS Status Structures) on_liveliness_changed
[DataWriterListener::on_liveliness_lost](#)([DataWriter](#) writer, [LivelinessLostStatus](#) DDS Status Structures) on_liveliness_lost

5.14 LoggingQosPolicy Class Reference

Controls the amount and kind of data that is logged.

5.14.1 Detailed Description

Controls the amount and kind of data that is logged.

The application must be linked with the instrumented library in order to enable logging with this [LoggingQosPolicy](#).

The level of log output is controlled by a series of bitmapped flags. Each class of log message is enabled by setting a flag to 1, and is disabled by 0.

Refer to `include/dds/dds.h` for the numeric values for the flags

5.15 OwnershipQosPolicy Class Reference

Determines instance ownership in the case of multiple writers. CoreDX DDS supports both SHARED_OWNERSHIP_QOS and EXCLUSIVE_OWNERSHIP_QOS.

5.15.1 Detailed Description

Determines instance ownership in the case of multiple writers. CoreDX DDS supports both SHARED_OWNERSHIP_QOS and EXCLUSIVE_OWNERSHIP_QOS.

5.16 OwnershipStrengthQosPolicy Class Reference

Defines the strength, or priority, of a Writer. The strength is used to determine ownership in the case of EXCLUSIVE_OWNERSHIP_QOS. When multiple writers publish data about the same instance, the **stronger** writer is considered the owner, and data from other writers is not delivered to the reader.

5.16.1 Detailed Description

Defines the strength, or priority, of a Writer. The strength is used to determine ownership in the case of EXCLUSIVE_OWNERSHIP_QOS. When multiple writers publish data about the same instance, the **stronger** writer is considered the owner, and data from other writers is not delivered to the reader.

5.17 PartitionQosPolicy Class Reference

Defines a logical data partition.

5.17.1 Detailed Description

Defines a logical data partition.

This QoS is assigned to a [Publisher](#) or [Subscriber](#). DataWriters and DataReaders exist within the partition defined by their parent. DataWriters and DataReaders communicate only if they have a matching partition (and all other QoS and topic parameters match).

The partition QoS can be changed dynamically. Dynamic changes to partition may cause new matches between DataWriters and DataReaders or may break existing matches.

The partition QoS comprises a vector of Strings. Two Partitions 'match' if any string in one matches any string in the other.

A partition string may be a regular expression with wildcards as defined by POSIX fnmatch API (1003.2-1992 section B.6).

An empty Partition, that is a Partition with an empty name vector, is treated as a Partition with a single empty string of "".

5.18 PeerParticipantQosPolicy Class Reference

Configures a list of [DomainParticipant](#) peers to attempt communication with.

Public Attributes

- Vector< [ParticipantLocator](#) > value
- boolean [strict_match](#)

5.18.1 Detailed Description

Configures a list of [DomainParticipant](#) peers to attempt communication with.

If the value list is empty, then default Discovery is used.

5.18.2 Member Data Documentation

5.18.2.1 boolean strict_match

restrict matching to those peers listed (default is 0/FALSE)

5.18.2.2 Vector<[ParticipantLocator](#)> value

list of Peer Locators with which we should attempt communication

5.19 PresentationQosPolicy Class Reference

Controls the presentation of received data samples to the application. CoreDX DDS currently supports only the `access_scope = INSTANCE_PRESENTATION_QOS` policy.

Public Attributes

- [PresentationQosPolicyAccessScopeKind access_scope](#)
the 'scope' of the presentation policy. Determines the extent to which the sample 'coherency' and 'order' are maintained.
- boolean [coherent_access](#)
Determines if coherent access is supported within the defined 'scope'.
- boolean [ordered_access](#)
Determines if ordered access is supported within the defined 'scope'.

5.19.1 Detailed Description

Controls the presentation of received data samples to the application. CoreDX DDS currently supports only the `access_scope = INSTANCE_PRESENTATION_QOS` policy.

5.20 PropertyQosPolicy Class Reference

The [PropertyQosPolicy](#) supports adding arbitrary name-value pairs.

Public Attributes

- `Vector< Property\_t > value`

5.20.1 Detailed Description

The [PropertyQosPolicy](#) supports adding arbitrary name-value pairs.

This can be used to configure non-standard properties in a portable fashion. In particular, this can be used to configure aspects of Security Plugin implementations.

5.20.2 Member Data Documentation

5.20.2.1 `Vector<Property_t> value`

Vector of properties.

5.21 ReaderDataLifecycleQosPolicy Class Reference

Specifies the lifecycle behavior of data instances managed by the [DataReader](#).

5.21.1 Detailed Description

Specifies the lifecycle behavior of data instances managed by the [DataReader](#).

autopurge_nowriter_samples_delay indicates how long the [DataReader](#) will maintain instance samples that have instance_state NOT_ALIVE_NO_WRITERS.

autopurge_disposed_samples_delay indicates how long the [DataReader](#) will retain information about instances that have instance_state NOT_ALIVE_DISPOSED.

5.22 ReliabilityQosPolicy Class Reference

Indicates the level of reliability offered/provided by the [Entity](#). If kind is RELIABLE_RELIABILITY_QOS, then the middleware will attempt to deliver all samples in the history cache. If samples are not received, then they will be retried.

Public Attributes

- [ReliabilityQosPolicyKind](#) kind
- [Duration_t](#) max_blocking_time

The amount of time to block a [DataWriter](#) if a write() or similar operation is unable to complete.

5.22.1 Detailed Description

Indicates the level of reliability offered/provided by the [Entity](#). If kind is RELIABLE_RELIABILITY_QOS, then the middleware will attempt to deliver all samples in the history cache. If samples are not received, then they will be retried.

A kind of BEST_EFFORT_RELIABILITY_QOS indicates that the middleware does not need to retry delivery of data samples.

The samples available in the history cache are affected by the [HistoryQosPolicy](#), [ResourceLimitsQosPolicy](#), and the [DurabilityQosPolicy](#).

Note: If samples are removed from the history cache, then they are not be available to be resent.

See also

[HistoryQosPolicy](#)
[ResourceLimitsQosPolicy](#)
[DurabilityQosPolicy](#)

5.22.2 Member Data Documentation

5.22.2.1 ReliabilityQosPolicyKind kind

Kind of Reliability

5.22.2.2 Duration_t max_blocking_time

The amount of time to block a [DataWriter](#) if a write() or similar operation is unable to complete.

For a [DataWriter](#), a write() or similar operation that attempts to add samples or instances to the data cache may be unable to complete under some [ReliabilityQosPolicy](#), [HistoryQosPolicy](#), and [ResourceLimitsQosPolicy](#) configurations. When this happens, the max_blocking_time will determine the amount of time the application should be blocked while attempting to complete the operation. If the operation is still unable to complete, an error is returned.

max_blocking_time may be 0, in which case an error is returned immediately if the operation is unable to complete.

5.23 ResourceLimitsQosPolicy Class Reference

Specifies the resources that the Service can use to maintain data samples and instances.

Public Attributes

- int [max_samples](#)
maximum samples allowed in the data cache
- int [max_instances](#)
maximum instances allowed in the cache
- int [max_samples_per_instance](#)
maximum samples per instance allowed in the cache
- boolean [preallocate_samples](#)
- boolean [preallocate_instances](#)

5.23.1 Detailed Description

Specifies the resources that the Service can use to maintain data samples and instances.

See also

[ReliabilityQosPolicy](#)
[HistoryQosPolicy](#)

5.23.2 Member Data Documentation

5.23.2.1 boolean [preallocate_instances](#)

Request that the Reader or Writer pre-allocate the specified number of instances (if [max_instances](#) is not unlimited).
CoreDX DDS Extension

5.23.2.2 boolean [preallocate_samples](#)

Request that the Reader or Writer pre-allocate the specified number of samples (if [max_samples](#) is not unlimited).
CoreDX DDS Extension

5.24 RpcQosPolicy Class Reference

Public Attributes

- String [service_instance_name](#)
- GUID_t [related_entity_guid](#)
- Vector< String > [topic_aliases](#)

5.24.1 Detailed Description

Augment a [DataWriter](#) or [DataReader](#) with RPC specific information.

5.24.2 Member Data Documentation

5.24.2.1 GUID_t related_entity_guid

guid of the related entity (request / reply)

5.24.2.2 String service_instance_name

the name of the service instance

5.24.2.3 Vector<String> topic_aliases

a sequence of aliases to support derived interfaces

5.25 RTPSReaderQosPolicy Class Reference

QoS Policy for configuring aspects of the RTPS Reader Protocol.

5.25.1 Detailed Description

QoS Policy for configuring aspects of the RTPS Reader Protocol.

5.26 RTPSWriterQosPolicy Class Reference

QoS Policy for configuring aspects of the RTPS Writer Protocol.

5.26.1 Detailed Description

QoS Policy for configuring aspects of the RTPS Writer Protocol.

5.27 ThreadModelQosPolicy Class Reference

QoS Policy for configuring the threading behavior of the [DomainParticipant](#).

5.27.1 Detailed Description

QoS Policy for configuring the threading behavior of the [DomainParticipant](#).

5.28 TimeBasedFilterQosPolicy Class Reference

Defines a filter based on time between samples. The [DataReader](#) indicates that it wants at most one sample for each instance every `minimum_separation` interval.

5.28.1 Detailed Description

Defines a filter based on time between samples. The [DataReader](#) indicates that it wants at most one sample for each instance every `minimum_separation` interval.

It is inconsistent for a [DataReader](#) to have a `minimum_separation` larger than its DEADLINE period. By default `minimum_separation=0` indicating that the [DataReader](#) should be sent all values.

5.29 TopicDataQosPolicy Class Reference

Allows the application to attach arbitrary information to a [Topic](#) QoS.

5.29.1 Detailed Description

Allows the application to attach arbitrary information to a [Topic](#) QoS.

The value of the TOPIC_DATA is propagated and made available to applications through the built-in topics.

The topic data is implemented as a vector of Bytes, and can be used to store any application defined data.

This QoS can be used by an application in combination with the [TopicListener](#) to implement conditional matching policies.

5.30 TransportPriorityQosPolicy Class Reference

A hint to the middleware to help configure the transport priority mechanism.

5.30.1 Detailed Description

A hint to the middleware to help configure the transport priority mechanism.

5.31 TypecodeQosPolicy Class Reference

Typecode representing the datatype a [DataReader](#) reads or a [DataWriter](#) writes.

5.31.1 Detailed Description

Typecode representing the datatype a [DataReader](#) reads or a [DataWriter](#) writes.

5.32 TypeConsistencyEnforcementQosPolicy Class Reference

This QoS policy establishes the policy for establishing consistency of the Reader and Writer data types.

Public Attributes

- [TypeConsistencyKind kind](#)

5.32.1 Detailed Description

This QoS policy establishes the policy for establishing consistency of the Reader and Writer data types.

The reader indicates the acceptable degree of type consistency.

5.32.2 Member Data Documentation

5.32.2.1 TypeConsistencyKind kind

Kind of Type Consistency Enforcement

5.33 UserDataQosPolicy Class Reference

Allows the application to attach arbitrary information to a [DomainParticipant](#), [DataWriter](#) or [DataReader](#).

5.33.1 Detailed Description

Allows the application to attach arbitrary information to a [DomainParticipant](#), [DataWriter](#) or [DataReader](#).

The value of the USER_DATA is propagated and made available to applications through the built-in topics.

The user data is implemented as a vector of Bytes, and can be used to store any application defined data.

This QoS can be used by an application in combination with the listeners to implement conditional matching policies.

5.34 WriterDataLifecycleQosPolicy Class Reference

Specifies the lifecycle behavior of data instances managed by the [DataWriter](#). If `autodispose_unregistered_instances` is true, then the [DataWriter](#) will automatically dispose any instances that are unregistered. **Note:** When a [DataWriter](#) is deleted, it will automatically unregister all of its instances. With this policy == true, then all instances will also be disposed.

5.34.1 Detailed Description

Specifies the lifecycle behavior of data instances managed by the [DataWriter](#). If `autodispose_unregistered_instances` is true, then the [DataWriter](#) will automatically dispose any instances that are unregistered. **Note:** When a [DataWriter](#) is deleted, it will automatically unregister all of its instances. With this policy == true, then all instances will also be disposed.

5.35 DestinationOrderQosPolicyKind Enum Reference

This enumeration contains the kinds of Destination Ordering.

Public Attributes

- [BY_RECEPTION_TIMESTAMP_DESTINATIONORDER_QOS](#)

5.35.1 Detailed Description

This enumeration contains the kinds of Destination Ordering.

5.35.2 Member Data Documentation

5.35.2.1 BY_RECEPTION_TIMESTAMP_DESTINATIONORDER_QOS

Use the reception time to determine the order of samples

5.36 DiscoveryQosPolicyDiscoveryKind Enum Reference

This enumeration contains the kinds of Discovery.

Public Attributes

- [PEER_DISCOVERY_QOS](#)

5.36.1 Detailed Description

This enumeration contains the kinds of Discovery.

5.36.2 Member Data Documentation

5.36.2.1 PEER_DISCOVERY_QOS

The [DomainParticipant](#) will use the standard peer-to-peer discovery

5.37 DurabilityQosPolicyKind Enum Reference

This enumeration contains the kinds of Durability.

Public Attributes

- [VOLATILE_DURABILITY_QOS](#)
- [TRANSIENT_LOCAL_DURABILITY_QOS](#)
- [TRANSIENT_DURABILITY_QOS](#)

5.37.1 Detailed Description

This enumeration contains the kinds of Durability.

5.37.2 Member Data Documentation

5.37.2.1 TRANSIENT_DURABILITY_QOS

The data is persisted beyond the lifecycle of the originating [DataWriter](#), and is available even after that [DataWriter](#) has been destroyed.

5.37.2.2 TRANSIENT_LOCAL_DURABILITY_QOS

The data is persisted locally within the source [DataWriter](#).

5.37.2.3 VOLATILE_DURABILITY_QOS

The data is volatile, and is not persisted beyond the initial publication.

5.38 HistoryQosPolicyKind Enum Reference

This enumeration contains the kinds of History.

Public Attributes

- [KEEP_LAST_HISTORY_QOS](#)

5.38.1 Detailed Description

This enumeration contains the kinds of History.

5.38.2 Member Data Documentation

5.38.2.1 KEEP_LAST_HISTORY_QOS

Store at most *depth* Samples in the data cache for each Instance. Samples may be overwritten

5.39 LivelinessQosPolicyKind Enum Reference

This enumeration contains the kinds of Liveliness.

Public Attributes

- [AUTOMATIC_LIVELINESS_QOS](#)
- [MANUAL_BY_PARTICIPANT_LIVELINESS_QOS](#)

5.39.1 Detailed Description

This enumeration contains the kinds of Liveliness.

5.39.2 Member Data Documentation

5.39.2.1 AUTOMATIC_LIVELINESS_QOS

The [DomainParticipant](#) automatically asserts liveliness for all DataWriters within the [DomainParticipant](#)

5.39.2.2 MANUAL_BY_PARTICIPANT_LIVELINESS_QOS

The application manually asserts the liveliness all DataWriters within this [DomainParticipant](#) by calling `assert_liveliness()` or writing data on any one [DataWriter](#) within this [DomainParticipant](#)

5.40 OwnershipQosPolicyKind Enum Reference

This enumeration contains the kinds of Ownership.

Public Attributes

- [SHARED_OWNERSHIP_QOS](#)

5.40.1 Detailed Description

This enumeration contains the kinds of Ownership.

5.40.2 Member Data Documentation

5.40.2.1 SHARED_OWNERSHIP_QOS

DataReaders receive data published by multiple DataWriters

5.41 PresentationQosPolicyAccessScopeKind Enum Reference

This enumeration contains the kinds of Access Scope for the [PresentationQosPolicy](#).

Public Attributes

- [INSTANCE_PRESENTATION_QOS](#)
- [TOPIC_PRESENTATION_QOS](#)

5.41.1 Detailed Description

This enumeration contains the kinds of Access Scope for the [PresentationQosPolicy](#).

5.41.2 Member Data Documentation

5.41.2.1 INSTANCE_PRESENTATION_QOS

The scope is within an Instance

5.41.2.2 TOPIC_PRESENTATION_QOS

The scope is within samples published by DataWriters on the [Topic](#)

5.42 ReliabilityQosPolicyKind Enum Reference

This enumeration contains the kinds of Reliability.

5.42.1 Detailed Description

This enumeration contains the kinds of Reliability.

5.43 TypeCodeKind Enum Reference

This enumeration contains the kinds of data types used in constructing a Typecode.

Static Public Member Functions

- static [TypeCodeKind coreToEnum](#) (DDS_TypeCodeKind k)

Public Attributes

- [TYPECODE_SHORT](#)
- [TYPECODE_LONG](#)
- [TYPECODE_USHORT](#)
- [TYPECODE_ULONG](#)
- [TYPECODE_FLOAT](#)
- [TYPECODE_DOUBLE](#)
- [TYPECODE_BOOLEAN](#)
- [TYPECODE_CHAR](#)
- [TYPECODE_OCTET](#)
- [TYPECODE_STRUCT](#)
- [TYPECODE_UNION](#)
- [TYPECODE_ENUM](#)
- [TYPECODE_STRING](#)
- [TYPECODE_SEQUENCE](#)
- [TYPECODE_ARRAY](#)
- [TYPECODE_ALIAS](#)
- [TYPECODE_LONGLONG](#)
- [TYPECODE_ULONGLONG](#)
- [TYPECODE_LONGDOUBLE](#)
- [TYPECODE_WCHAR](#)
- [TYPECODE_WSTRING](#)
- [TYPECODE_VALUE](#)

5.43.1 Detailed Description

This enumeration contains the kinds of data types used in constructing a Typecode.

5.43.2 Member Function Documentation

5.43.2.1 `static TypeCodeKind coreToEnum ( DDS_TypeCodeKind k ) [static]`

< Not used. Included for compatibility.

5.43.3 Member Data Documentation

5.43.3.1 `TYPECODE_ALIAS`

Not used. Included for compatibility.

5.43.3.2 TYPECODE_ARRAY

Typecode to identify an 'array' data type.

5.43.3.3 TYPECODE_BOOLEAN

Typecode to identify a 'boolean' data type

5.43.3.4 TYPECODE_CHAR

Typecode to identify a 'char' data type (8 bits)

5.43.3.5 TYPECODE_DOUBLE

Typecode to identify a 'double' data type

5.43.3.6 TYPECODE_ENUM

Typecode to identify an 'enum' data type

5.43.3.7 TYPECODE_FLOAT

Typecode to identify a 'float' data type

5.43.3.8 TYPECODE_LONG

Typecode to identify a 'long' data type (32 bits with sign)

5.43.3.9 TYPECODE_LONGDOUBLE

Typecode to identify a 'long double' data type. Not supported by CoreDX DDS.

5.43.3.10 TYPECODE_LONGLONG

Typecode to identify a 'long long' data type (64 bits with sign)

5.43.3.11 TYPECODE_OCTET

Typecode to identify an 'octet' data type (8 bits)

5.43.3.12 TYPECODE_SEQUENCE

Typecode to identify a 'sequence' data type. May be bounded length or unlimited.

5.43.3.13 TYPECODE_SHORT

Typecode to identify a 'short' data type (16 bits with sign)

5.43.3.14 TYPECODE_STRING

Typecode to identify a 'string' data type. May be bounded length or unlimited.

5.43.3.15 TYPECODE_STRUCT

Typecode to identify a 'struct' data type (with named sub-fields)

5.43.3.16 TYPECODE_ULONG

Typecode to identify an 'unsigned long' data type (32 bits without sign)

5.43.3.17 TYPECODE_ULONGLONG

Typecode to identify an 'unsigned long long' data type (64 bits without sign)

5.43.3.18 TYPECODE_UNION

Typecode to identify a 'union' data type (with named sub-fields, selected by a discriminator)

5.43.3.19 TYPECODE_USHORT

Typecode to identify an 'unsigned short' data type (16 bits without sign)

5.43.3.20 TYPECODE_VALUE

Not used. Included for compatibility.

5.43.3.21 TYPECODE_WCHAR

Typecode to identify a 'wide char' data type. Not supported by CoreDX DDS.

5.43.3.22 TYPECODE_WSTRING

Typecode to identify a 'wide string' data type. Not supported by CoreDX DDS.

5.44 TypeConsistencyKind Enum Reference

This enumeration contains the kinds of Reliability.

Public Attributes

- [DISALLOW_TYPE_COERCION](#)
- [ALLOW_TYPE_COERCION](#)

5.44.1 Detailed Description

This enumeration contains the kinds of Reliability.

5.44.2 Member Data Documentation

5.44.2.1 ALLOW_TYPE_COERCION

allow coercion

5.44.2.2 DISALLOW_TYPE_COERCION

disallow coercion (type name and structure must match)

Chapter 6

QoS Provider

6.1 QosProvider Class Reference

[QosProvider](#) loads QoS settings from a library and provides interfaces to access entity specific QoS Policies.

Public Member Functions

- [QosProvider](#) (String uri, String profile)
Constructor. Creates a new [QosProvider](#) instance that loads QoS settings from a library.
- void [cleanup](#) ()
"Destructor". Returns a [QosProvider](#) back to the factory.
- [DomainParticipantFactoryQos get_participantfactory_qos](#) (String id)
Access a [DomainParticipantFactory](#) QoS.
- [DomainParticipantQos get_participant_qos](#) (String id)
Access a [DomainParticipant](#) QoS.
- [TopicQos get_topic_qos](#) (String id)
Access a [Topic](#) QoS.
- [SubscriberQos get_subscriber_qos](#) (String id)
Access a [Subscriber](#) QoS.
- [PublisherQos get_publisher_qos](#) (String id)
Access a [Publisher](#) QoS.
- [DataReaderQos get_datareader_qos](#) (String id)
Access a [DataReader](#) QoS.
- [DataWriterQos get_datawriter_qos](#) (String id)
Access a [DataWriter](#) QoS.

6.1.1 Detailed Description

[QosProvider](#) loads QoS settings from a library and provides interfaces to access entity specific QoS Policies.

Chapter 7

DDS Listeners

7.1 DataReaderListener Interface Reference

The [DataReaderListener](#) provides asynchronous notification of [DataReader](#) events. This listener can be installed during [DataReader](#) creation, `DomainParticipant_create_yyy()`, as well as by calling `DataReader_set_listener()`.

Inherited by `Replier< TReq extends RequestType, TRep extends ReplyType >.ReplierDRListener`, and `ServiceProxy< TReq extends RequestType, TRep extends ReplyType >.ServiceProxyDRListener`.

Public Member Functions

- void [on_requested_deadline_missed](#) ([DataReader](#) the_reader, [RequestedDeadlineMissedStatus](#) status)
- void [on_requested_incompatible_qos](#) ([DataReader](#) the_reader, [RequestedIncompatibleQosStatus](#) status)
- void [on_sample_rejected](#) ([DataReader](#) the_reader, [SampleRejectedStatus](#) status)
- void [on_liveliness_changed](#) ([DataReader](#) the_reader, [LivelinessChangedStatus](#) status)
- void [on_data_available](#) ([DataReader](#) the_reader)
- void [on_subscription_matched](#) ([DataReader](#) the_reader, [SubscriptionMatchedStatus](#) status)
- void [on_sample_lost](#) ([DataReader](#) the_reader, [SampleLostStatus](#) status)
- long [get_nil_mask](#) ()

7.1.1 Detailed Description

The [DataReaderListener](#) provides asynchronous notification of [DataReader](#) events. This listener can be installed during [DataReader](#) creation, `DomainParticipant_create_yyy()`, as well as by calling `DataReader_set_listener()`.

7.1.2 Member Function Documentation

7.1.2.1 long [get_nil_mask](#) ()

[get_nil_mask\(\)](#) returns a bitmask indicating which listener methods (if any) should be considered NIL, and therefore, should not be invoked.

7.1.2.2 void [on_data_available](#) ([DataReader](#) *the_reader*)

Called when the CoreDX DDS middleware detects that new data or data state information is available.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.1.2.3 void on_liveliness_changed (*DataReader the_reader*, *LivelinessChangedStatus status*)

Called when the CoreDX DDS middleware detects that the liveliness of a matched [DataWriter](#) has changed (either 'active' or 'inactive').

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.1.2.4 void on_requested_deadline_missed (*DataReader the_reader*, *RequestedDeadlineMissedStatus status*)

Called when the CoreDX DDS middleware detects that the deadline specified in the [DataReader](#) QoS DEADLINE policy was not satisfied for a data instance.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.1.2.5 void on_requested_incompatible_qos (*DataReader the_reader*, *RequestedIncompatibleQosStatus status*)

Called when the CoreDX DDS middleware detects that the [DataReader](#) requested a QoS policy that was incompatible with that offered by a [DataWriter](#).

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.1.2.6 void on_sample_lost (*DataReader the_reader*, *SampleLostStatus status*)

Called when the CoreDX DDS middleware detects that a sample has been lost (never received).

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.1.2.7 void on_sample_rejected (*DataReader the_reader*, *SampleRejectedStatus status*)

Called when the CoreDX DDS middleware detects that a sample has been rejected.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.1.2.8 void on_subscription_matched (*DataReader the_reader*, *SubscriptionMatchedStatus status*)

Called when the CoreDX DDS middleware detects that the [DataReader](#) has matched or ceased to be matched with a [DataWriter](#).

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.2 DataWriterListener Interface Reference

The [DataWriterListener](#) provides asynchronous notification of [DataWriter](#) events. This listener can be installed during [DataWriter](#) creation, `Publisher_create_datawriter()`, as well as by calling `DataWriter_set_listener()`.

Inherited by `ServiceProxy` < `TReq` extends `RequestType`, `TRep` extends `ReplyType` >. `ServiceProxyDWListener`.

Public Member Functions

- void `on_offered_deadline_missed` ([DataWriter](#) writer, [OfferedDeadlineMissedStatus](#) status)
- void `on_offered_incompatible_qos` ([DataWriter](#) writer, [OfferedIncompatibleQosStatus](#) status)
- void `on_liveliness_lost` ([DataWriter](#) writer, [LivelinessLostStatus](#) status)
- void `on_publication_matched` ([DataWriter](#) writer, [PublicationMatchedStatus](#) status)
- long `get_nil_mask` ()

7.2.1 Detailed Description

The [DataWriterListener](#) provides asynchronous notification of [DataWriter](#) events. This listener can be installed during [DataWriter](#) creation, `Publisher_create_datawriter()`, as well as by calling `DataWriter_set_listener()`.

7.2.2 Member Function Documentation

7.2.2.1 long `get_nil_mask` ()

`get_nil_mask()` returns a bitmask indicating which listener methods (if any) should be considered NIL, and therefore, should not be invoked.

7.2.2.2 void `on_liveliness_lost` ([DataWriter](#) writer, [LivelinessLostStatus](#) status)

Called when the CoreDX DDS middleware detects that the [DataWriter](#) has not satisfied its LIVELINESS QoS setting.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.2.2.3 void `on_offered_deadline_missed` ([DataWriter](#) writer, [OfferedDeadlineMissedStatus](#) status)

Called when the CoreDX DDS middleware detects that the deadline that the `DDS_DataWriter` has offered Through the DEADLINE QoS was not satisfied. That is, the [DataWriter](#) has failed to update an instance with the frequency specified in the DEADLINE QoS.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.2.2.4 void `on_offered_incompatible_qos` ([DataWriter](#) writer, [OfferedIncompatibleQosStatus](#) status)

Called when the CoreDX DDS middleware detects that the [DataWriter](#) has offered a QoS policy setting that is incompatible with that requested by a potentially matching [DataReader](#).

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.2.2.5 void on_publication_matched (**DataWriter** *writer*, **PublicationMatchedStatus** *status*)

Called when the CoreDX DDS middleware detects that the [DataWriter](#) has matched with a [DataReader](#) or has ceased to be matched with a [DataReader](#).

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3 DomainParticipantListener Interface Reference

Public Member Functions

- void `on_inconsistent_topic` (`Topic` the_topic, `InconsistentTopicStatus` status)
- void `on_offered_deadline_missed` (`DataWriter` writer, `OfferedDeadlineMissedStatus` status)
- void `on_offered_incompatible_qos` (`DataWriter` writer, `OfferedIncompatibleQosStatus` status)
- void `on_liveliness_lost` (`DataWriter` writer, `LivelinessLostStatus` status)
- void `on_publication_matched` (`DataWriter` writer, `PublicationMatchedStatus` status)
- void `on_requested_deadline_missed` (`DataReader` the_reader, `RequestedDeadlineMissedStatus` status)
- void `on_requested_incompatible_qos` (`DataReader` the_reader, `RequestedIncompatibleQosStatus` status)
- void `on_sample_rejected` (`DataReader` the_reader, `SampleRejectedStatus` status)
- void `on_liveliness_changed` (`DataReader` the_reader, `LivelinessChangedStatus` status)
- void `on_data_available` (`DataReader` the_reader)
- void `on_subscription_matched` (`DataReader` the_reader, `SubscriptionMatchedStatus` status)
- void `on_sample_lost` (`DataReader` the_reader, `SampleLostStatus` status)
- void `on_data_on_readers` (`Subscriber` the_subscriber)
- long `get_nil_mask` ()

7.3.1 Detailed Description

The `DomainParticipantListener` provides asynchronous notification of DDS `DomainParticipant` events. This listener can be installed during `DomainParticipant` creation, `DomainParticipantFactory_create_participant()`, as well as by calling `DomainParticipant_set_listener()`.

7.3.2 Member Function Documentation

7.3.2.1 `long get_nil_mask ( )`

`get_nil_mask()` returns a bitmask indicating which listener methods (if any) should be considered NIL, and therefore, should not be invoked.

7.3.2.2 `void on_data_available ( DataReader the_reader )`

Called when the CoreDX DDS middleware detects that a `DataReader`, contained in any `Subscriber` in this `DomainParticipant`, has new data or data state information available. This listener is invoked only if the concerned `DataReader` and `Subscriber` do not have an `on_data_available` listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.3 `void on_data_on_readers ( Subscriber the_subscriber )`

Called when the CoreDX DDS middleware detects that a `DataReader`, contained in any `Subscriber` in this `DomainParticipant`, has new data or data state information available. This listener is invoked only if the concerned `Subscriber` does not have an `on_data_on_readers` listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.4 void on_inconsistent_topic (Topic *the_topic*, InconsistentTopicStatus *status*)

Called when the CoreDX DDS middleware detects that a [Topic](#) contained within this [DomainParticipant](#) has characteristics that are different (inconsistent) with another existing topic. This listener is invoked only if the concerned [Topic](#) does not have an **on_inconsistent_topic** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.5 void on_liveliness_changed (DataReader *the_reader*, LivelinessChangedStatus *status*)

Called when the CoreDX DDS middleware detects that the liveliness of a [DataWriter](#), matched to a [DataReader](#) within any [Subscriber](#) within this [DomainParticipant](#), has changed (either 'active' or 'inactive'). This listener is invoked only if the concerned [DataReader](#) and [Subscriber](#) do not have an **on_liveliness_changed** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.6 void on_liveliness_lost (DataWriter *writer*, LivelinessLostStatus *status*)

Called when the CoreDX DDS middleware detects that a [DataWriter](#) contained in any [Publisher](#) within this [DomainParticipant](#) has not satisfied its LIVELINESS QoS setting. This listener is invoked only if the concerned [DataWriter](#) and [Publisher](#) do not have an **on_liveliness_lost** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.7 void on_offered_deadline_missed (DataWriter *writer*, OfferedDeadlineMissedStatus *status*)

Called when the CoreDX DDS middleware detects that a [DataWriter](#) contained in any [Publisher](#) within this [DomainParticipant](#) has failed to meet its DEADLINE QoS commitment. This listener is invoked only if the concerned [DataWriter](#) and [Publisher](#) do not have an **on_offered_deadline_missed** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.8 void on_offered_incompatible_qos (DataWriter *writer*, OfferedIncompatibleQosStatus *status*)

Called when the CoreDX DDS middleware detects that a [DataWriter](#) contained in any [Publisher](#) within this [DomainParticipant](#) has offered a QoS policy setting that is incompatible with that requested by a potentially matching [DataReader](#). This listener is invoked only if the concerned [DataWriter](#) and [Publisher](#) do not have an **on_offered_incompatible_qos** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.9 void on_publication_matched (DataWriter *writer*, PublicationMatchedException *status*)

Called when the CoreDX DDS middleware detects that a [DataWriter](#) contained in any [Publisher](#) within this [DomainParticipant](#) has matched with a [DataReader](#) or has ceased to be matched with a [DataReader](#). This listener is invoked only if the concerned [DataWriter](#) and [Publisher](#) do not have an **on_publication_matched** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.10 void on_requested_deadline_missed (DataReader *the_reader*, RequestedDeadlineMissedStatus *status*)

Called when the CoreDX DDS middleware detects that the QoS DEADLINE policy of a [DataReader](#), contained in any [Subscriber](#) of this [DomainParticipant](#), was not satisfied for a data instance. This listener is invoked only if the concerned

[DataReader](#) and [Subscriber](#) do not have an **on_requested_deadline_missed** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.11 void on_requested_incompatible_qos ([DataReader](#) *the_reader*, [RequestedIncompatibleQosStatus](#) *status*)

Called when the CoreDX DDS middleware detects that a [DataReader](#) contained in any [Subscriber](#) within this [DomainParticipant](#), has requested a QoS policy that was incompatible with that offered by a [DataWriter](#). This listener is invoked only if the concerned [DataReader](#) and [Subscriber](#) do not have an **on_requested_incompatible_qos** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.12 void on_sample_lost ([DataReader](#) *the_reader*, [SampleLostStatus](#) *status*)

Called when the CoreDX DDS middleware detects that a [DataReader](#), contained in any [Subscriber](#) in this [DomainParticipant](#), has lost a sample (never received). This listener is invoked only if the concerned [DataReader](#) and [Subscriber](#) do not have an **on_sample_lost** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.13 void on_sample_rejected ([DataReader](#) *the_reader*, [SampleRejectedStatus](#) *status*)

Called when the CoreDX DDS middleware detects that a [DataReader](#) contained in any [Subscriber](#) within the DomainParticipant has rejected a sample. This listener is invoked only if the concerned [DataReader](#) and [Subscriber](#) do not have an **on_sample_rejected** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.3.2.14 void on_subscription_matched ([DataReader](#) *the_reader*, [SubscriptionMatchedException](#) *status*)

Called when the CoreDX DDS middleware detects that a [DataReader](#), contained in any [Subscriber](#) in this [DomainParticipant](#), has matched or ceased to be matched with a [DataWriter](#). This listener is invoked only if the concerned [DataReader](#) and [Subscriber](#) do not have an **on_subscription_matched** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.4 PublisherListener Interface Reference

Public Member Functions

- void `on_offered_deadline_missed` (`DataWriter` writer, `OfferedDeadlineMissedStatus` status)
- void `on_offered_incompatible_qos` (`DataWriter` writer, `OfferedIncompatibleQosStatus` status)
- void `on_liveliness_lost` (`DataWriter` writer, `LivelinessLostStatus` status)
- void `on_publication_matched` (`DataWriter` writer, `PublicationMatchedStatus` status)
- long `get_nil_mask` ()

7.4.1 Detailed Description

The `PublisherListener` provides asynchronous notification of `Publisher` events. This listener can be installed during `Publisher` creation `DomainParticipant_create_publisher()`, as well as by calling `Publisher_set_listener()`.

7.4.2 Member Function Documentation

7.4.2.1 long `get_nil_mask` ()

`get_nil_mask()` returns a bitmask indicating which listener methods (if any) should be considered NIL, and therefore, should not be invoked.

7.4.2.2 void `on_liveliness_lost` (`DataWriter` writer, `LivelinessLostStatus` status)

Called when the CoreDX DDS middleware detects that a `DataWriter` contained in the `Publisher` has not satisfied its LIVELINESS QoS setting. This listener is invoked only if the concerned `DataWriter` does not have an `on_offered_deadline_missed` listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.4.2.3 void `on_offered_deadline_missed` (`DataWriter` writer, `OfferedDeadlineMissedStatus` status)

Called when the CoreDX DDS middleware detects that a `DataWriter` contained in the `Publisher` has failed to meet its DEADLINE QoS commitment. This listener is invoked only if the concerned `DataWriter` does not have an `on_offered_deadline_missed` listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.4.2.4 void `on_offered_incompatible_qos` (`DataWriter` writer, `OfferedIncompatibleQosStatus` status)

Called when the CoreDX DDS middleware detects that a `DataWriter` contained in the `Publisher` has offered a QoS policy setting that is incompatible with that requested by a potentially matching `DataReader`. This listener is invoked only if the concerned `DataWriter` does not have an `on_offered_deadline_missed` listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.4.2.5 void `on_publication_matched` (`DataWriter` writer, `PublicationMatchedStatus` status)

Called when the CoreDX DDS middleware detects that a `DataWriter` contained in the `Publisher` has matched with a `DataReader` or has ceased to be matched with a `DataReader`. This listener is invoked only if the concerned `DataWriter`

does not have an **on_offered_deadline_missed** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.5 SubscriberListener Interface Reference

The [SubscriberListener](#) provides asynchronous notification of [Subscriber](#) events. This listener can be installed during [Subscriber](#) creation, `DomainParticipant_create_subscriber()` as well as by calling `Subscriber_set_listener()`.

Public Member Functions

- void `on_requested_deadline_missed` ([DataReader](#) the_reader, [RequestedDeadlineMissedStatus](#) status)
- void `on_requested_incompatible_qos` ([DataReader](#) the_reader, [RequestedIncompatibleQosStatus](#) status)
- void `on_sample_rejected` ([DataReader](#) the_reader, [SampleRejectedStatus](#) status)
- void `on_liveliness_changed` ([DataReader](#) the_reader, [LivelinessChangedStatus](#) status)
- void `on_data_available` ([DataReader](#) the_reader)
- void `on_subscription_matched` ([DataReader](#) the_reader, [SubscriptionMatchStatus](#) status)
- void `on_sample_lost` ([DataReader](#) the_reader, [SampleLostStatus](#) status)
- void `on_data_on_readers` ([Subscriber](#) the_subscriber)
- long `get_nil_mask` ()

7.5.1 Detailed Description

The [SubscriberListener](#) provides asynchronous notification of [Subscriber](#) events. This listener can be installed during [Subscriber](#) creation, `DomainParticipant_create_subscriber()` as well as by calling `Subscriber_set_listener()`.

7.5.2 Member Function Documentation

7.5.2.1 long `get_nil_mask` ()

`get_nil_mask()` returns a bitmask indicating which listener methods (if any) should be considered NIL, and therefore, should not be invoked.

7.5.2.2 void `on_data_available` ([DataReader](#) the_reader)

Called when the CoreDX DDS middleware detects that a [DataReader](#) contained in the [Subscriber](#) has new data or data state information available. This listener is invoked only if the concerned [DataReader](#) does not have an **on_data_available** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.5.2.3 void `on_data_on_readers` ([Subscriber](#) the_subscriber)

Called when the CoreDX DDS middleware detects that data or data state information arrives at any [DataReader](#) contained within the [Subscriber](#).

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.5.2.4 void `on_liveliness_changed` ([DataReader](#) the_reader, [LivelinessChangedStatus](#) status)

Called when the CoreDX DDS middleware detects that the liveliness of a [DataWriter](#), matched to a [DataReader](#) within this [Subscriber](#), has changed (either 'active' or 'inactive'). This listener is invoked only if the concerned [DataReader](#) does not have an **on_liveliness_changed** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.5.2.5 void on_requested_deadline_missed (DataReader *the_reader*, RequestedDeadlineMissedStatus *status*)

Called when the CoreDX DDS middleware detects that the QoS DEADLINE policy of a [DataReader](#), contained in this [Subscriber](#), was not satisfied for a data instance. This listener is invoked only if the concerned [DataReader](#) does not have an **on_requested_deadline_missed** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.5.2.6 void on_requested_incompatible_qos (DataReader *the_reader*, RequestedIncompatibleQosStatus *status*)

[on_requested_incompatible_qos\(\)](#) is called when the CoreDX DDS middleware detects that a [DataReader](#) contained in the [Subscriber](#) has requested a QoS policy that was incompatible with that offered by a [DataWriter](#). This listener is invoked only if the concerned [DataReader](#) does not have an **on_requested_incompatible_qos** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.5.2.7 void on_sample_lost (DataReader *the_reader*, SampleLostStatus *status*)

Called when the CoreDX DDS middleware detects that a [DataReader](#) contained in the [Subscriber](#) has lost a sample (never received). This listener is invoked only if the concerned [DataReader](#) does not have an **on_sample_lost** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.5.2.8 void on_sample_rejected (DataReader *the_reader*, SampleRejectedStatus *status*)

Called when the CoreDX DDS middleware detects that a [DataReader](#) contained in the [Subscriber](#) has rejected a sample. This listener is invoked only if the concerned [DataReader](#) does not have an **on_sample_rejected** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.5.2.9 void on_subscription_matched (DataReader *the_reader*, SubscriptionMatchedStatus *status*)

Called when the CoreDX DDS middleware detects that a [DataReader](#) contained in the [Subscriber](#) has matched or ceased to be matched with a [DataWriter](#). This listener is invoked only if the concerned [DataReader](#) does not have an **on_subscription_matched** listener installed.

The **status** argument provides a snapshot of the status at the time the listener was invoked.

7.6 TopicListener Interface Reference

The DDS_TopicListener provides asynchronous notification of DDS_Topic events. This listener can be installed during [Topic](#) creation (DDS_DomainParticipant_create_topic() and related) as well as by calling DDS_Topic_set_listener().

Public Member Functions

- void [on_inconsistent_topic](#) ([Topic](#) the_topic, [InconsistentTopicStatus](#) status)
- long [get_nil_mask](#) ()

7.6.1 Detailed Description

The DDS_TopicListener provides asynchronous notification of DDS_Topic events. This listener can be installed during [Topic](#) creation (DDS_DomainParticipant_create_topic() and related) as well as by calling DDS_Topic_set_listener().

7.6.2 Member Function Documentation

7.6.2.1 long get_nil_mask ()

Returns a bitmask indicating which listener methods (if any) should be considered NIL, and therefore, should not be invoked.

7.6.2.2 void on_inconsistent_topic ([Topic](#) *the_topic*, [InconsistentTopicStatus](#) *status*)

Called when the CoreDX DDS middleware detects that another topic exists with different (inconsistent) characteristics. The **status** argument provides a snapshot of the status at the time the listener was invoked.

Chapter 8

DDS Status Structures

These data types are used to communicate status information in the Listener and `get_XYZ_status()` methods.

8.1 InconsistentTopicStatus Class Reference

[InconsistentTopicStatus](#) provides status related to the `on_inconsistent_topic` listener methods of the [TopicListener](#) structure.

Public Member Functions

- `int` [get_total_count](#) ()
- `int` [get_total_count_change](#) ()

8.1.1 Detailed Description

[InconsistentTopicStatus](#) provides status related to the `on_inconsistent_topic` listener methods of the [TopicListener](#) structure.

8.1.2 Member Function Documentation

8.1.2.1 `int` [get_total_count](#) ()

Gets the **total_count** value. Cumulative count of the discovered Topics having a matching name and inconsistent characteristics.

8.1.2.2 `int` [get_total_count_change](#) ()

Gets the **total_count_change** value. Change in **total_count** since the last time the listener was called or status was read.

8.2 LivelinessChangedStatus Class Reference

Status related to the `on_liveliness_changed` listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

Public Member Functions

- [int get_alive_count \(\)](#)
- [int get_alive_count_change \(\)](#)
- [int get_not_alive_count \(\)](#)
- [int get_not_alive_count_change \(\)](#)
- [InstanceHandle_t get_last_publication_handle \(\)](#)

8.2.1 Detailed Description

Status related to the `on_liveliness_changed` listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

8.2.2 Member Function Documentation

8.2.2.1 `int get_alive_count ( )`

Get the number of 'active' DataWriters matched to this [DataReader](#).

8.2.2.2 `int get_alive_count_change ( )`

Get the change in **alive_count** since the last time the listener was called or status was read.

8.2.2.3 `InstanceHandle_t get_last_publication_handle ( )`

Get the handle identifying the most recent [DataWriter](#) whose liveliness changed.

8.2.2.4 `int get_not_alive_count ( )`

Get the number of 'not-alive' DataWriters matched to this [DataReader](#).

8.2.2.5 `int get_not_alive_count_change ( )`

Get the change in **not_alive_count** since the last time the listener was called or status was read.

8.3 LivelinessLostStatus Class Reference

Status related to the on_liveliness_lost listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.

Public Member Functions

- int [get_total_count](#) ()
- int [get_total_count_change](#) ()

8.3.1 Detailed Description

Status related to the on_liveliness_lost listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.

8.3.2 Member Function Documentation

8.3.2.1 int [get_total_count](#) ()

Get the cumulative number of times that an 'alive' [DataWriter](#) became not alive.

8.3.2.2 int [get_total_count_change](#) ()

Get the change in **total_count** since the last time the listener was called or status was read.

8.4 OfferedDeadlineMissedStatus Class Reference

Status related to the `on_offered_deadline_missed` listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.

Public Member Functions

- [int get_total_count\(\)](#)
- [int get_total_count_change\(\)](#)
- [InstanceHandle_t get_last_instance_handle\(\)](#)

8.4.1 Detailed Description

Status related to the `on_offered_deadline_missed` listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.

8.4.2 Member Function Documentation

8.4.2.1 `InstanceHandle_t get_last_instance_handle()`

Get the handle identifying the most recent instance whose deadline was missed.

8.4.2.2 `int get_total_count()`

Get the cumulative count of the number of deadlines missed by this [DataWriter](#).

8.4.2.3 `int get_total_count_change()`

Get the change in **total_count** since the last time the listener was called or status was read.

8.5 OfferedIncompatibleQoSStatus Class Reference

Status related to the `on_offered_incompatible_qos` listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.

Public Member Functions

- `int` [get_total_count](#) ()
- `int` [get_total_count_change](#) ()
- `int` [get_last_policy_id](#) ()
- `Vector` [get_policies](#) ()

8.5.1 Detailed Description

Status related to the `on_offered_incompatible_qos` listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.

8.5.2 Member Function Documentation

8.5.2.1 `int` [get_last_policy_id](#) ()

Get the **id** of the most recent requested incompatible QoS policy. Convert **id** to a descriptive string with `DDS.qos_policy_str(int qos_policy_id)`.

8.5.2.2 `Vector` [get_policies](#) ()

A list of QoS policies and the total number of times each QoS policy was found to be incompatible. Vector elements are of type [QoSPolicyCount](#).

8.5.2.3 `int` [get_total_count](#) ()

Get the cumulative count of the number of DataWriters discovered having matching [Topic](#) and incompatible QoS.

8.5.2.4 `int` [get_total_count_change](#) ()

Get the change in **total_count** since the last time the listener was called or status was read.

8.6 PublicationMatchedStatus Class Reference

Status related to the `on_publication_matched` listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.

Public Member Functions

- [int get_total_count \(\)](#)
- [int get_total_count_change \(\)](#)
- [int get_current_count \(\)](#)
- [int get_current_count_change \(\)](#)
- [InstanceHandle_t get_last_subscription_handle \(\)](#)

8.6.1 Detailed Description

Status related to the `on_publication_matched` listener methods of the [DataWriter](#), [Publisher](#), and [DomainParticipant](#) structures.

8.6.2 Member Function Documentation

8.6.2.1 `int get_current_count ( )`

Get the current number of DataReaders matched to the [DataWriter](#).

8.6.2.2 `int get_current_count_change ( )`

Get the change in **current_count** since the last time the listener was called or status was read.

8.6.2.3 `InstanceHandle_t get_last_subscription_handle ( )`

Get the handle of the most recently matched [DataReader](#) (subscription).

8.6.2.4 `int get_total_count ( )`

Get the cumulative count of the number of times this [DataWriter](#) has discovered a matching [DataReader](#).

8.6.2.5 `int get_total_count_change ( )`

Get the change in **total_count** since the last time the listener was called or status was read.

8.7 RequestedDeadlineMissedStatus Class Reference

Status related to the on_requested_deadline_missed listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

Public Member Functions

- int [get_total_count](#) ()
- int [get_total_count_change](#) ()
- [InstanceHandle_t](#) [get_last_instance_handle](#) ()

8.7.1 Detailed Description

Status related to the on_requested_deadline_missed listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

8.7.2 Member Function Documentation

8.7.2.1 [InstanceHandle_t](#) [get_last_instance_handle](#) ()

Get the handle identifying the most recent instance whose deadline was missed.

8.7.2.2 int [get_total_count](#) ()

Get the cumulative count of the number of detected deadline misses for any instance read by the [DataReader](#).

8.7.2.3 int [get_total_count_change](#) ()

Get the change in **total_count** since the last time the listener was called or status was read.

8.8 RequestedIncompatibleQosStatus Class Reference

Public Member Functions

- int [get_total_count](#) ()
- int [get_total_count_change](#) ()
- int [get_last_policy_id](#) ()
- Vector [get_policies](#) ()

8.8.1 Detailed Description

Status related to the `on_requested_incompatible_qos` listener methods of the `DDS_DataReader`, `DDS_Subscriber`, and `DDS_DomainParticipant` structures.

8.8.2 Member Function Documentation

8.8.2.1 int [get_last_policy_id](#) ()

Get the **id** identifying the most recent QoS policy detected to be incompatible. Convert **id** to a descriptive string with `DDS.qos_policy_str(int qos_policy_id)`.

8.8.2.2 Vector [get_policies](#) ()

Get a list of QoS policies and the total number of times each QoS policy was found to be incompatible. Vector elements are of type [QosPolicyCount](#).

8.8.2.3 int [get_total_count](#) ()

Get the cumulative count of the number of discovered DataReaders having matching [Topic](#) and incompatible QoS.

8.8.2.4 int [get_total_count_change](#) ()

Get the change in **total_count** since the last time the listener was called or status was read.

8.9 SampleLostStatus Class Reference

Status related to the on_sample_lost listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

Public Member Functions

- int [get_total_count](#) ()
- int [get_total_count_change](#) ()

8.9.1 Detailed Description

Status related to the on_sample_lost listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

8.9.2 Member Function Documentation

8.9.2.1 int [get_total_count](#) ()

Gets the **total_count**. Cumulative count of all samples lost under the [Topic](#).

8.9.2.2 int [get_total_count_change](#) ()

Gets the **total_count_change**. Change in **total_count** since the last time the listener was called or status was read.

8.10 SampleRejectedStatus Class Reference

Status related to the `on_sample_rejected` listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

Public Member Functions

- `int get_total_count ()`
- `int get_total_count_change ()`
- `SampleRejectedStatusKind get_last_reason ()`
- `InstanceHandle_t get_last_instance_handle ()`

8.10.1 Detailed Description

Status related to the `on_sample_rejected` listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

8.10.2 Member Function Documentation

8.10.2.1 `InstanceHandle_t get_last_instance_handle ( )`

Get the handle of the instance associated with the most recently rejected sample.

8.10.2.2 `SampleRejectedStatusKind get_last_reason ( )`

Get the reason for rejecting the most recently rejected sample.

8.10.2.3 `int get_total_count ( )`

Get the cumulative count of all samples rejected under the [Topic](#).

8.10.2.4 `int get_total_count_change ( )`

Get the change in **total_count** since the last time the listener was called or status was read.

8.11 SubscriptionMatchedStatus Class Reference

Status related to the on_subscription_matched listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

Public Member Functions

- int [get_total_count](#) ()
- int [get_total_count_change](#) ()
- int [get_current_count](#) ()
- int [get_current_count_change](#) ()
- [InstanceHandle_t](#) [get_last_publication_handle](#) ()

8.11.1 Detailed Description

Status related to the on_subscription_matched listener methods of the [DataReader](#), [Subscriber](#), and [DomainParticipant](#) structures.

8.11.2 Member Function Documentation

8.11.2.1 int get_current_count ()

Get the current number of DataWriters matched to the [DataReader](#).

8.11.2.2 int get_current_count_change ()

Get the change in **current_count** since the last time the listener was called or status was read.

8.11.2.3 InstanceHandle_t get_last_publication_handle ()

Get the handle of most recent matched [DataWriter](#) (publication).

8.11.2.4 int get_total_count ()

Get the cumulative count of the number of times this [DataReader](#) has discovered a matching [DataWriter](#).

8.11.2.5 int get_total_count_change ()

Get the change in **total_count** since the last time the listener was called or status was read.

Chapter 9

DDS Samples

The data types documented in this section are currently used only by the RPC API.

9.1 `LoanedSamples< T >` Class Template Reference

Public Member Functions

- `int length ()`
- `int size ()`
- `void return_loan ()`
- `void release (DataReaderI< T > reader, SampleSeq< T > data_seq, SampleInfoSeq info_seq)`

9.1.1 Detailed Description

Holds a collection of sample data and related meta-data.

The `LoanedSamples` object holds the loaned data and meta-data returned from a `DataReader::read()` operation. The object manages the 'loan' by automatically calling `return_loan()` on the source `DataReader` when the object is destroyed.

9.1.2 Member Function Documentation

9.1.2.1 `int length ( )`

Access the length of the contained data sequence.

9.1.2.2 `void release ( DataReaderI< T > reader, SampleSeq< T > data_seq, SampleInfoSeq info_seq )`

Transfers the loan to the caller.

The `LoanedSamples` instance will no longer manage the loaned data. The caller is responsible for ensuring `DataReader::return_loan()` is invoked as required.

9.1.2.3 `void return_loan ( )`

Return the loaned data to the originating `DataReader`.

After this call, the [LoanedSamples](#) object no longer manages the data or sample info sequences.

9.1.2.4 int size ()

Access the length of the contained data sequence.

Chapter 10

Builtin DDS Types

10.1 DomainId_t Class Reference

10.1.1 Detailed Description

Identifies a DDS Domain. The [DomainParticipant](#) is a member of a Domain as identified by the DomainId provided at creation time. Domains are represented as integers.

10.2 `Duration_t` Class Reference

Represents a time duration with nanosecond resolution.

Public Member Functions

- [Duration_t](#) (int s, long n)
Constructor.
- [Duration_t](#) (DDS_Duration_t d)
Constructor.
- void [subtract](#) ([Duration_t](#) other)
Subtract 'other' from this.

10.2.1 Detailed Description

Represents a time duration with nanosecond resolution.

10.3 InstanceHandle_t Class Reference

Used to identify a DDS [Entity](#) or data instance. An InstanceHandle is unique within its context. For example, each data instance known by a [DataReader](#) is assigned a unique InstanceHandle. However, these handles may not be unique when compared to handles from another [DataReader](#).

10.3.1 Detailed Description

Used to identify a DDS [Entity](#) or data instance. An InstanceHandle is unique within its context. For example, each data instance known by a [DataReader](#) is assigned a unique InstanceHandle. However, these handles may not be unique when compared to handles from another [DataReader](#).

10.4 Property_t Class Reference

A name-value pair property. The 'propagate' flag indicates if this property should be transferred through discovery.

Public Attributes

- String [name](#)
- String [value](#)
- boolean [propagate](#)

10.4.1 Detailed Description

A name-value pair property. The 'propagate' flag indicates if this property should be transferred through discovery.

10.4.2 Member Data Documentation

10.4.2.1 String name

The property name

10.4.2.2 boolean propagate

indicate if property should be conveyed via Discovery

10.4.2.3 String value

The property value

10.5 Time_t Class Reference

Represents time with nanosecond resolution.

10.5.1 Detailed Description

Represents time with nanosecond resolution.

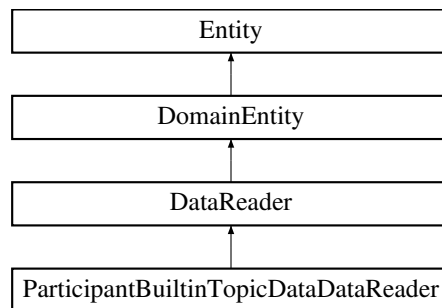
Chapter 11

DDS Discovery Types

11.1 ParticipantBuiltinTopicDataDataReader Class Reference

The [ParticipantBuiltinTopicDataDataReader](#) is a built-in [DataReader](#) that can be used to access Participant Discovery information. The reader can be accessed through the special 'builtin' [Subscriber](#) via the [DomainParticipant.get_builtin_subscriber\(\)](#) routine.

Inheritance diagram for ParticipantBuiltinTopicDataDataReader:



Additional Inherited Members

11.1.1 Detailed Description

The [ParticipantBuiltinTopicDataDataReader](#) is a built-in [DataReader](#) that can be used to access Participant Discovery information. The reader can be accessed through the special 'builtin' [Subscriber](#) via the [DomainParticipant.get_builtin_subscriber\(\)](#) routine.

See also

- [FooDataReader](#)
- [ParticipantBuiltinTopicData](#)
- [DomainParticipant::get_builtin_subscriber](#)

11.2 ParticipantBuiltinTopicData Class Reference

The [ParticipantBuiltinTopicData](#) is a built-in Data Type that represents Participant Discovery information.

11.2.1 Detailed Description

The [ParticipantBuiltinTopicData](#) is a built-in Data Type that represents Participant Discovery information.

The specialized reader ([ParticipantBuiltinTopicDataReader](#)) can be accessed through the special 'builtin' [Subscriber](#) via the [DomainParticipant.get_builtin_subscriber\(\)](#) routine.

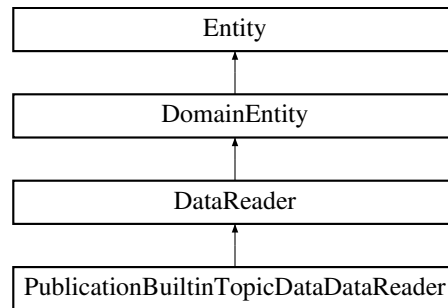
See also

[ParticipantBuiltinTopicDataReader](#)
[DomainParticipant::get_builtin_subscriber](#)

11.3 PublicationBuiltinTopicDataDataReader Class Reference

The [PublicationBuiltinTopicDataDataReader](#) is a built-in [DataReader](#) that can be used to access Publication Discovery information. The reader can be accessed through the special 'builtin' [Subscriber](#) via the [DomainParticipant.get_builtin_subscriber\(\)](#) routine.

Inheritance diagram for [PublicationBuiltinTopicDataDataReader](#):



Additional Inherited Members

11.3.1 Detailed Description

The [PublicationBuiltinTopicDataDataReader](#) is a built-in [DataReader](#) that can be used to access Publication Discovery information. The reader can be accessed through the special 'builtin' [Subscriber](#) via the [DomainParticipant.get_builtin_subscriber\(\)](#) routine.

See also

- [FooDataReader](#)
- [PublicationBuiltinTopicData](#)
- [DomainParticipant::get_builtin_subscriber](#)

11.4 PublicationBuiltinTopicData Class Reference

The [PublicationBuiltinTopicData](#) is a built-in Data Type that represents Publication Discovery information.

11.4.1 Detailed Description

The [PublicationBuiltinTopicData](#) is a built-in Data Type that represents Publication Discovery information.

The specialized reader ([PublicationBuiltinTopicDataReader](#)) can be accessed through the special 'builtin' [Subscriber](#) via the [DomainParticipant.get_builtin_subscriber\(\)](#) routine.

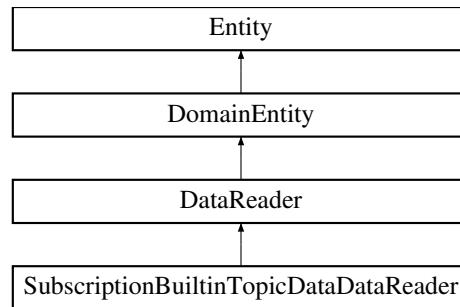
See also

[SubscriptionBuiltinTopicDataReader](#)
[DomainParticipant::get_builtin_subscriber](#)

11.5 SubscriptionBuiltinTopicDataDataReader Class Reference

The [SubscriptionBuiltinTopicDataDataReader](#) is a built-in [DataReader](#) that can be used to access Subscription Discovery information.

Inheritance diagram for SubscriptionBuiltinTopicDataDataReader:



Additional Inherited Members

11.5.1 Detailed Description

The [SubscriptionBuiltinTopicDataDataReader](#) is a built-in [DataReader](#) that can be used to access Subscription Discovery information.

The reader can be accessed through the special 'builtin' [Subscriber](#) via the [DomainParticipant.get_builtin_subscriber\(\)](#) routine.

See also

[FooDataReader](#)
[SubscriptionBuiltinTopicData](#)
[DomainParticipant::get_builtin_subscriber](#)

11.6 SubscriptionBuiltinTopicData Class Reference

The [SubscriptionBuiltinTopicData](#) is a built-in Data Type that represents Subscription Discovery information.

11.6.1 Detailed Description

The [SubscriptionBuiltinTopicData](#) is a built-in Data Type that represents Subscription Discovery information.

The specialized reader ([SubscriptionBuiltinTopicDataReader](#)) can be accessed through the special 'builtin' [Subscriber](#) via the [DomainParticipant.get_builtin_subscriber\(\)](#) routine.

See also

[SubscriptionBuiltinTopicDataReader](#)
[DomainParticipant::get_builtin_subscriber](#)

Chapter 12

Supporting Types

12.1 Constants Class Reference

Provides various constants and utility routines used throughout CoreDX DDS.

Static Public Member Functions

- static String [qos_policy_str](#) (int qos_policy_id)

Static Public Attributes

- static final int [READ_SAMPLE_STATE](#) = (0x0001)
- static final int [NOT_READ_SAMPLE_STATE](#) = (0x0002)
- static final int [ANY_SAMPLE_STATE](#) = (0x00FF)
- static final int [NEW_VIEW_STATE](#) = (0x0001)
- static final int [NOT_NEW_VIEW_STATE](#) = (0x0002)
- static final int [ANY_VIEW_STATE](#) = (0x00FF)
- static final int [ALIVE_INSTANCE_STATE](#) = (0x0001)
- static final int [NOT_ALIVE_DISPOSED_INSTANCE_STATE](#) = (0x0002)
- static final int [NOT_ALIVE_NO_WRITERS_INSTANCE_STATE](#) = (0x0004)
- static final int [NOT_ALIVE_INSTANCE_STATE](#) = (0x0006)
- static final int [ANY_INSTANCE_STATE](#) = (0x00FF)
- static final int [INCONSISTENT_TOPIC_STATUS](#) = (0x0001)
- static final int [OFFERED_DEADLINE_MISSED_STATUS](#) = (0x0002)
- static final int [REQUESTED_DEADLINE_MISSED_STATUS](#) = (0x0004)
- static final int [OFFERED_INCOMPATIBLE_QOS_STATUS](#) = (0x0008)
- static final int [REQUESTED_INCOMPATIBLE_QOS_STATUS](#) = (0x0010)
- static final int [SAMPLE_LOST_STATUS](#) = (0x0020)
- static final int [SAMPLE_REJECTED_STATUS](#) = (0x0040)
- static final int [DATA_ON_READERS_STATUS](#) = (0x0080)
- static final int [DATA_AVAILABLE_STATUS](#) = (0x0100)
- static final int [LIVELINESS_LOST_STATUS](#) = (0x0200)
- static final int [LIVELINESS_CHANGED_STATUS](#) = (0x0400)
- static final int [PUBLICATION_MATCHED_STATUS](#) = (0x0800)
- static final int [SUBSCRIPTION_MATCHED_STATUS](#) = (0x1000)

- static final int [NO_STATUS](#) = (0)
- static final int [STATUS_MASK_NONE](#) = (0)
- static final int [ALL_STATUS](#) = (0xFFFF)
- static final int [USERDATA_QOS_POLICY_ID](#) = 1
- static final int [DURABILITY_QOS_POLICY_ID](#) = 2
- static final int [PRESENTATION_QOS_POLICY_ID](#) = 3
- static final int [DEADLINE_QOS_POLICY_ID](#) = 4
- static final int [LATENCYBUDGET_QOS_POLICY_ID](#) = 5
- static final int [OWNERSHIP_QOS_POLICY_ID](#) = 6
- static final int [OWNERSHIPSTRENGTH_QOS_POLICY_ID](#) = 7
- static final int [LIVELINESS_QOS_POLICY_ID](#) = 8
- static final int [TIMEBASEDFILTER_QOS_POLICY_ID](#) = 9
- static final int [PARTITION_QOS_POLICY_ID](#) = 10
- static final int [RELIABILITY_QOS_POLICY_ID](#) = 11
- static final int [DESTINATIONORDER_QOS_POLICY_ID](#) = 12
- static final int [HISTORY_QOS_POLICY_ID](#) = 13
- static final int [RESOURCELIMITS_QOS_POLICY_ID](#) = 14
- static final int [ENTITYFACTORY_QOS_POLICY_ID](#) = 15
- static final int [WRITERDATALIFECYCLE_QOS_POLICY_ID](#) = 16
- static final int [READERDATALIFECYCLE_QOS_POLICY_ID](#) = 17
- static final int [TOPICDATA_QOS_POLICY_ID](#) = 18
- static final int [GROUPDATA_QOS_POLICY_ID](#) = 19
- static final int [TRANSPORTPRIORITY_QOS_POLICY_ID](#) = 20
- static final int [LIFESPAN_QOS_POLICY_ID](#) = 21
- static final int [DURABILITYSERVICE_QOS_POLICY_ID](#) = 22
- static final int [DATA_REPRESENTATION_QOS_POLICY_ID](#) = 23

12.1.1 Detailed Description

Provides various constants and utility routines used throughout CoreDX DDS.

[Constants](#) found in this class:

Quality of Service [Constants](#)

- [xxx_QOS_POLICY_ID](#): Represents a specific QoS Policy

Miscellaneous [Constants](#)

- [HANDLE_xxx](#): Represents 'HANDLE_NIL' (or no handle) as defined by the DDS standards.
- [LENGTH_ULIMITED](#): Represents an unlimited length, used for read calls, for example.

State [Constants](#)

- [xxx_STATE](#): Used in the [DataReader](#) for read calls and conditions to indicate types of data

Status [Constants](#):

- [xxx_STATUS](#): Used by [Condition](#) and Listeners to describe statuses and events that may occur within the middle-ware.

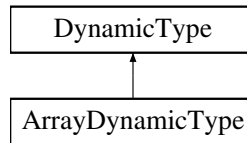
Chapter 13

X-Types DynamicType API

13.1 ArrayDynamicType Class Reference

[ArrayDynamicType](#) holds an 'Array' value.

Inheritance diagram for ArrayDynamicType:



Additional Inherited Members

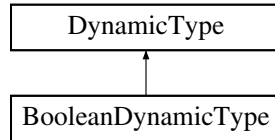
13.1.1 Detailed Description

[ArrayDynamicType](#) holds an 'Array' value.

13.2 BooleanDynamicType Class Reference

[BooleanDynamicType](#) holds an 'boolean' value.

Inheritance diagram for BooleanDynamicType:



Additional Inherited Members

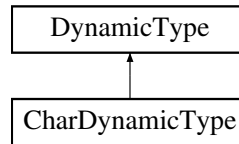
13.2.1 Detailed Description

[BooleanDynamicType](#) holds an 'boolean' value.

13.3 CharDynamicType Class Reference

[CharDynamicType](#) holds an 'char' value.

Inheritance diagram for CharDynamicType:



Additional Inherited Members

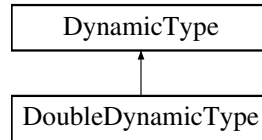
13.3.1 Detailed Description

[CharDynamicType](#) holds an 'char' value.

13.4 DoubleDynamicType Class Reference

[DoubleDynamicType](#) holds an 'double' value.

Inheritance diagram for DoubleDynamicType:



Additional Inherited Members

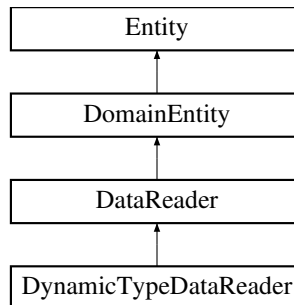
13.4.1 Detailed Description

[DoubleDynamicType](#) holds an 'double' value.

13.5 DynamicTypeDataReader Class Reference

[DynamicTypeDataReader](#) is an implementation of [DataReader](#) that supports DynamicTypes.

Inheritance diagram for DynamicTypeDataReader:



Additional Inherited Members

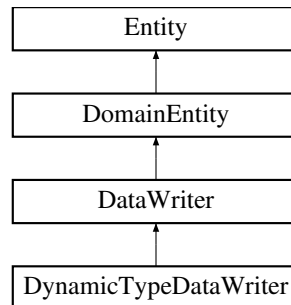
13.5.1 Detailed Description

[DynamicTypeDataReader](#) is an implementation of [DataReader](#) that supports DynamicTypes.

13.6 DynamicTypeDataWriter Class Reference

[DynamicTypeDataWriter](#) is an implementation of [DataWriter](#) that supports DynamicTypes.

Inheritance diagram for DynamicTypeDataWriter:



Additional Inherited Members

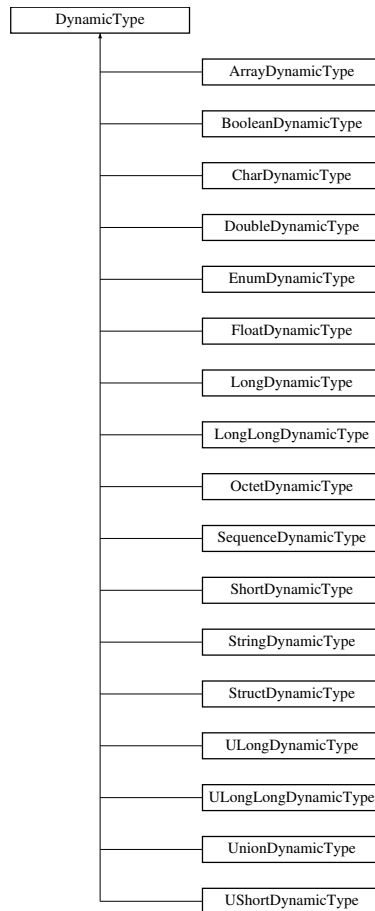
13.6.1 Detailed Description

[DynamicTypeDataWriter](#) is an implementation of [DataWriter](#) that supports DynamicTypes.

13.7 DynamicType Class Reference

[DynamicType](#) is the base class for all specific DynamicTypes.

Inheritance diagram for DynamicType:



Public Member Functions

- void [release](#) ()

13.7.1 Detailed Description

[DynamicType](#) is the base class for all specific DynamicTypes.

13.7.2 Member Function Documentation

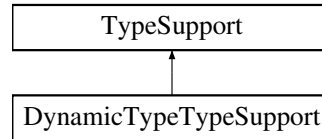
13.7.2.1 void release ()

Release the underlying dynamic type data.

13.8 DynamicTypeTypeSupport Class Reference

[DynamicTypeTypeSupport](#) is an implementation of [TypeSupport](#) that supports DynamicTypes.

Inheritance diagram for DynamicTypeTypeSupport:



Public Member Functions

- `ReturnCode_t register_type (DomainParticipant dp, String type_name)`
- `long getCTypeSupport ()`
- `DataReader create_datareader (Subscriber sub, TopicDescription td, SWIGTYPE_p__DataReader jni_dr)`
- `DataWriter create_datawriter (Publisher pub, Topic topic, SWIGTYPE_p__DataWriter jni_dw)`

Static Public Member Functions

- static `DynamicTypeTypeSupport create_dynamic_typesupport (TypecodeQosPolicy typecode)`
Create a [TypeSupport](#) based on a discovered TypeCode. Using the builtin [PublicationBuiltinTopicDataDataReader](#) or [SubscriptionBuiltinTopicDataDataReader](#) it is possible for an application to discover external Entities and their associated data types. With the typecode that defines a data type, it is possible to generate a [TypeSupport](#) object that is capable of supporting DataReaders and DataWriters of this discovered data type. This allows an application to discover and interact with entities within the domain without prior knowledge of the data types.

13.8.1 Detailed Description

[DynamicTypeTypeSupport](#) is an implementation of [TypeSupport](#) that supports DynamicTypes.

13.8.2 Member Function Documentation

13.8.2.1 DataReader create_datareader (Subscriber sub, TopicDescription td, SWIGTYPE_p__DataReader jni_dr)

Private.

Implements [TypeSupport](#).

13.8.2.2 DataWriter create_datawriter (Publisher pub, Topic topic, SWIGTYPE_p__DataWriter jni_dw)

Private.

Implements [TypeSupport](#).

13.8.2.3 long getCTypeSupport ()

Private.

Implements [TypeSupport](#).

13.8.2.4 `ReturnCode_t register_type ( DomainParticipant dp, String type_name )`

Registers a new Application Defined Data Type with the DDS infrastructure. This method is implemented by the application specific [TypeSupport](#) class generated by `coredx_ddl`.

The type is registered under the name 'type_name'. If the 'type_name' parameter is null, then the default name will be used.

See also

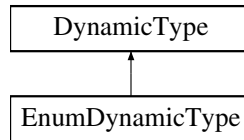
[com.toc.coredx.DDS.FooTypeSupport FooTypeSupport](#)

Implements [TypeSupport](#).

13.9 EnumDynamicType Class Reference

[EnumDynamicType](#) represents ENUMs.

Inheritance diagram for EnumDynamicType:



Additional Inherited Members

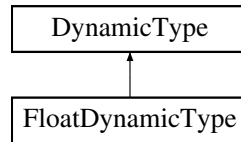
13.9.1 Detailed Description

[EnumDynamicType](#) represents ENUMs.

13.10 FloatDynamicType Class Reference

[FloatDynamicType](#) holds an 'float' value.

Inheritance diagram for FloatDynamicType:



Additional Inherited Members

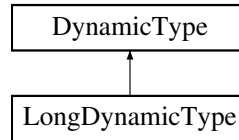
13.10.1 Detailed Description

[FloatDynamicType](#) holds an 'float' value.

13.11 LongDynamicType Class Reference

[LongDynamicType](#) holds an 'long' value (32bits, signed).

Inheritance diagram for LongDynamicType:



Additional Inherited Members

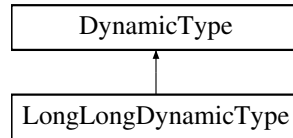
13.11.1 Detailed Description

[LongDynamicType](#) holds an 'long' value (32bits, signed).

13.12 LongLongDynamicType Class Reference

[LongLongDynamicType](#) holds an 'long long' (64bit, signed) value.

Inheritance diagram for LongLongDynamicType:



Additional Inherited Members

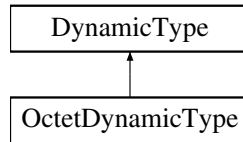
13.12.1 Detailed Description

[LongLongDynamicType](#) holds an 'long long' (64bit, signed) value.

13.13 OctetDynamicType Class Reference

[OctetDynamicType](#) holds an 'octet' value.

Inheritance diagram for OctetDynamicType:



Additional Inherited Members

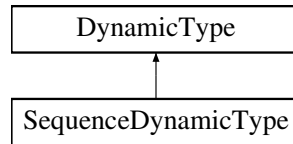
13.13.1 Detailed Description

[OctetDynamicType](#) holds an 'octet' value.

13.14 SequenceDynamicType Class Reference

[SequenceDynamicType](#) holds an 'Sequence' value.

Inheritance diagram for SequenceDynamicType:



Additional Inherited Members

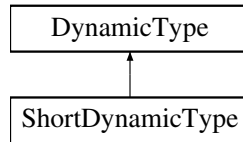
13.14.1 Detailed Description

[SequenceDynamicType](#) holds an 'Sequence' value.

13.15 ShortDynamicType Class Reference

[ShortDynamicType](#) holds an 'short' value.

Inheritance diagram for ShortDynamicType:



Additional Inherited Members

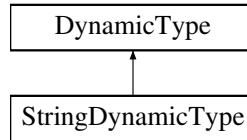
13.15.1 Detailed Description

[ShortDynamicType](#) holds an 'short' value.

13.16 StringDynamicType Class Reference

[StringDynamicType](#) holds an 'string' value.

Inheritance diagram for StringDynamicType:



Additional Inherited Members

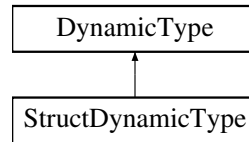
13.16.1 Detailed Description

[StringDynamicType](#) holds an 'string' value.

13.17 StructDynamicType Class Reference

[StructDynamicType](#) represents structures.

Inheritance diagram for StructDynamicType:



Additional Inherited Members

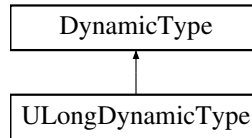
13.17.1 Detailed Description

[StructDynamicType](#) represents structures.

13.18 ULongDynamicType Class Reference

[ULongDynamicType](#) holds an 'unsigned long' value (32bits, unsigned).

Inheritance diagram for ULongDynamicType:



Additional Inherited Members

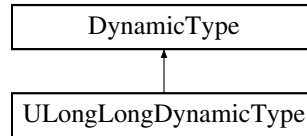
13.18.1 Detailed Description

[ULongDynamicType](#) holds an 'unsigned long' value (32bits, unsigned).

13.19 ULongLongDynamicType Class Reference

[ULongLongDynamicType](#) holds an 'long long' (64bit, unsigned) value.

Inheritance diagram for ULongLongDynamicType:



Additional Inherited Members

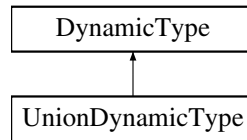
13.19.1 Detailed Description

[ULongLongDynamicType](#) holds an 'long long' (64bit, unsigned) value.

13.20 UnionDynamicType Class Reference

[UnionDynamicType](#) represents unionures.

Inheritance diagram for UnionDynamicType:



Additional Inherited Members

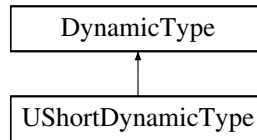
13.20.1 Detailed Description

[UnionDynamicType](#) represents unionures.

13.21 UShortDynamicType Class Reference

[UShortDynamicType](#) holds an 'unsigned short' value.

Inheritance diagram for UShortDynamicType:



Additional Inherited Members

13.21.1 Detailed Description

[UShortDynamicType](#) holds an 'unsigned short' value.

Chapter 14

CoreDX DDS Transports

14.1 IpTransportInterface Class Reference

Defines an interface for use by the [UdpTransport](#).

14.1.1 Detailed Description

Defines an interface for use by the [UdpTransport](#).

See also

[UdpTransportConfig](#)

14.2 LmtTransportConfig Class Reference

Structure that holds LMT [Transport](#) configuration items.

Public Member Functions

- [LmtTransportConfig](#) ()
Constructor. Initializes all configuration items with default values.
- `ReturnCode_t` [get_default_config](#) ()
- `ReturnCode_t` [get_env_config](#) ()

Public Attributes

- `int` [so_sndbuf](#)
- `int` [so_rcvbuf](#)
- `int` [max_tx_size](#)
- `int` [max_rx_buf_size](#)
- `int` [debug_flags](#)

14.2.1 Detailed Description

Structure that holds LMT [Transport](#) configuration items.

See also

[DomainParticipant::add_transport\(Transport\)](#) `add_transport()`

14.2.2 Member Function Documentation

14.2.2.1 `ReturnCode_t get_default_config ( )`

Initialize the [LmtTransportConfig](#) object with default values. Currently assigned values may be overwritten by defaults.

14.2.2.2 `ReturnCode_t get_env_config ( )`

Query for environment variables that impact lmt transport configuration. Load the values (if any) into the [LmtTransportConfig](#) object. Currently assigned values may be overwritten by values derived from environment variables.

14.2.3 Member Data Documentation

14.2.3.1 `int debug_flags`

configure debug output of the transport

14.2.3.2 `int max_rx_buf_size`

limit size of RX buffer (per connection)

14.2.3.3 int max_tx_size

largest LMT packet size we transmit (default: 8192, max: 64K)

14.2.3.4 int so_rcvbuf

size in bytes for socket RCVBUF

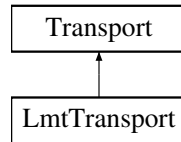
14.2.3.5 int so_sndbuf

size in bytes for socket SNDBUF

14.3 LmtTransport Class Reference

An instance of a LMT [Transport](#) that can be added to a [DomainParticipant](#).

Inheritance diagram for LmtTransport:



Static Public Member Functions

- static [Transport](#) `create_transport` ([LmtTransportConfig](#) transport_config)
Constructor.

14.3.1 Detailed Description

An instance of a LMT [Transport](#) that can be added to a [DomainParticipant](#).

The [LmtTransport](#) support RTPS communication through a local machine mechanism for optimized on-platform communications.

See also

[DomainParticipant::add_transport\(Transport\)](#) `add_transport()`

14.4 TcpTransportConfig Class Reference

Structure that holds TCP [Transport](#) configuration items.

Public Member Functions

- [TcpTransportConfig](#) ()
Constructor. Initializes all configuration items with default values.
- `ReturnCode_t` [get_default_config](#) ()
- `ReturnCode_t` [get_env_config](#) ()

Public Attributes

- `short` [participant_index](#)
- `Vector< IpTransportInterface >` [interfaces](#)
- `boolean` [dynamic_interfaces](#)
- `int` [tx_max_packet_size](#)

14.4.1 Detailed Description

Structure that holds TCP [Transport](#) configuration items.

See also

[DomainParticipant::add_transport\(Transport\)](#) [add_transport\(\)](#)

14.4.2 Member Function Documentation

14.4.2.1 `ReturnCode_t` [get_default_config](#) ()

Initialize the [TcpTransportConfig](#) object with default values. Currently assigned values may be overwritten by defaults.

14.4.2.2 `ReturnCode_t` [get_env_config](#) ()

Query for environment variables that impact tcp transport configuration. Load the values (if any) into the [TcpTransportConfig](#) object. Currently assigned values may be overwritten by values derived from environment variables.

14.4.3 Member Data Documentation

14.4.3.1 `boolean` [dynamic_interfaces](#)

detect and handle changes to interface addresses

14.4.3.2 `Vector<IpTransportInterface>` [interfaces](#)

default: empty -> use all available interfaces

14.4.3.3 short participant_index

-1: auto detect; else force (may fail if another participant is using the ports (can't exceed 120)

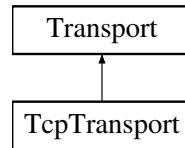
14.4.3.4 int tx_max_packet_size

default: 64K (udp limit)

14.5 TcpTransport Class Reference

An instance of a TCP [Transport](#) that can be added to a [DomainParticipant](#).

Inheritance diagram for TcpTransport:



Static Public Member Functions

- static [Transport](#) [create_transport](#) ([TcpTransportConfig](#) transport_config)
Constructor.

14.5.1 Detailed Description

An instance of a TCP [Transport](#) that can be added to a [DomainParticipant](#).

The [TcpTransport](#) support RTPS over a TCP/IP network. It can be used for both on-platform and off-platform communications.

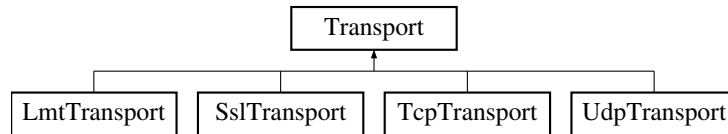
See also

[DomainParticipant::add_transport\(Transport\)](#) [add_transport\(\)](#)

14.6 Transport Class Reference

An instance of a [Transport](#) that can be added to a [DomainParticipant](#).

Inheritance diagram for Transport:



14.6.1 Detailed Description

An instance of a [Transport](#) that can be added to a [DomainParticipant](#).

The [UdpTransport](#) is the default transport used by CoreDX DDS for communications. It support RTPS over a UDP/IP network. It can be used for both on-platform and off-platform communications.

See also

[DomainParticipant::add_transport\(Transport\)](#) `add_transport()`

14.7 UdpTransportConfig Class Reference

Structure that holds UDP [Transport](#) configuration items.

Public Member Functions

- [UdpTransportConfig](#) ()
Constructor. Initializes all configuration items with default values.
- `ReturnCode_t` [get_default_config](#) ()
- `ReturnCode_t` [get_env_config](#) ()

Public Attributes

- `short` [participant_index](#)
- `boolean` [use_ipv4](#)
- `boolean` [use_ipv6](#)
- `Vector< IpTransportInterface >` [interfaces](#)
- `boolean` [dynamic_interfaces](#)
- `int` [rx_init_buffer_size](#)
- `int` [rx_max_buffer_size](#)
- `int` [tx_max_packet_size](#)
- `int` [so_rcvbuf](#)
- `int` [so_sndbuf](#)
- `byte` [meta_multicast_address_v4](#) []
- `byte` [user_multicast_address_v4](#) []
- `byte` [meta_multicast_address_v6](#) []
- `byte` [user_multicast_address_v6](#) []
- `byte` [multicast_ttl](#)
- `boolean` [tx_meta_multicast](#)
- `boolean` [tx_meta_unicast](#)
- `boolean` [rx_meta_multicast](#)
- `boolean` [rx_user_multicast](#)
- `boolean` [advertise_meta_multicast](#)
- `boolean` [advertise_user_multicast](#)
- `byte` [broadcast_address](#) []
- `boolean` [do_meta_broadcast](#)

14.7.1 Detailed Description

Structure that holds UDP [Transport](#) configuration items.

See also

[DomainParticipant::add_transport\(Transport\)](#) `add_transport()`

14.7.2 Member Function Documentation

14.7.2.1 `ReturnCode_t get_default_config ( )`

Initialize the [UdpTransportConfig](#) object with default values. Currently assigned values may be overwritten by defaults.

14.7.2.2 `ReturnCode_t get_env_config ( )`

Query for environment variables that impact udp transport configuration. Load the values (if any) into the [UdpTransport-Config](#) object. Currently assigned values may be overwritten by values derived from environment variables.

14.7.3 Member Data Documentation

14.7.3.1 `boolean advertise_meta_multicast`

advertise we can RX META MULTICAST

14.7.3.2 `boolean advertise_user_multicast`

advertise we can RX USER MULTICAST

14.7.3.3 `byte broadcast_address[]`

4 byte IPv4 address. Default: 255.255.255.255

14.7.3.4 `boolean do_meta_broadcast`

enable broadcast of META (DPD discovery) data default: 0 (off)

14.7.3.5 `boolean dynamic_interfaces`

detect and handle changes to interface addresses

14.7.3.6 `Vector<IpTransportInterface> interfaces`

default: empty -> use all available interfaces

14.7.3.7 `byte meta_multicast_address_v4[]`

4 byte IPv4 address for meta (discovery) traffic. Default: [239 255 0 1] per the standard

14.7.3.8 `byte meta_multicast_address_v6[]`

16 byte IPv6 address for meta (discovery) traffic. Default: [ff03:0000:0000:0000:0000:0000:0000:0001]

14.7.3.9 `byte multicast_ttl`

default: 1 (0: disable all MCAST TX)

14.7.3.10 `short participant_index`

-1: auto detect; else force (may fail if another participant is using the ports (can't exceed 120))

14.7.3.11 int rx_init_buffer_size

initial size of data buffer

14.7.3.12 int rx_max_buffer_size

maximum size of data buffer

14.7.3.13 boolean rx_meta_multicast

enable META MULTICAST (discovery) RX

14.7.3.14 boolean rx_user_multicast

enable USER MULTICAST (data) RX

14.7.3.15 int so_rcvbuf

socket RCVBUF size (set to -1 to use OS default)

14.7.3.16 int so_sndbuf

socket SNDBUF size (set to -1 to use OS default)

14.7.3.17 int tx_max_packet_size

default: 64K (udp limit)

14.7.3.18 boolean tx_meta_multicast

enable META MULTICAST (discovery) TX

14.7.3.19 boolean tx_meta_unicast

enable META UNICAST (discovery) TX

14.7.3.20 boolean use_ipv4

Support IPv4 communications (default ON (1))

14.7.3.21 boolean use_ipv6

Support IPv4 communications (default OFF(0))

14.7.3.22 byte user_multicast_address_v4[]

4 byte IPv4 address for user traffic. Default: [239 255 0 1] per the standard

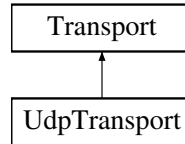
14.7.3.23 byte user_multicast_address_v6[]

16 byte IPv6 address for user traffic. Default: [ff03:0000:0000:0000:0000:0000:0000:0001]

14.8 UdpTransport Class Reference

An instance of a [Transport](#) that can be added to a [DomainParticipant](#).

Inheritance diagram for UdpTransport:



Static Public Member Functions

- static [Transport](#) [create_transport](#) ([UdpTransportConfig](#) transport_config)
Constructor.

14.8.1 Detailed Description

An instance of a [Transport](#) that can be added to a [DomainParticipant](#).

The [UdpTransport](#) is the default transport used by CoreDX DDS for communications. It support RTPS over a UDP/IP network. It can be used for both on-platform and off-platform communications.

See also

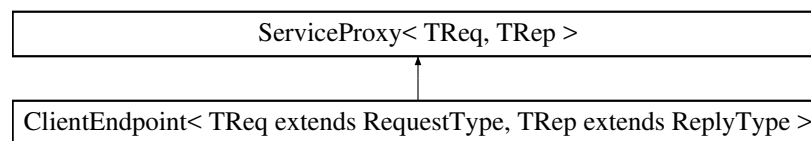
[DomainParticipant::add_transport\(Transport\)](#) [add_transport\(\)](#)

Chapter 15

CoreDX DDS RPC over DDS API

15.1 ClientEndpoint< TReq extends RequestType, TRep extends ReplyType > Class Template Reference

Inheritance diagram for ClientEndpoint< TReq extends RequestType, TRep extends ReplyType >:



Public Member Functions

- [ClientEndpoint](#) (Class< TReq > req_class, Class< TRep > rep_class, [ClientParams](#) cparams)
- [ClientParams](#) get_client_params ()

Protected Member Functions

- ReturnCode_t [send_request](#) (TReq request)
- boolean [receive_reply](#) (Sample< TRep > reply, SampleIdentity_t relatedRequestId)

15.1.1 Detailed Description

A [ClientEndpoint](#) provides functions to obtain the underlying DDS entities at the client side. [ClientEndpoint](#) inherits from [ServiceProxy](#). A [ClientEndpoint](#) shall not be instantiated directly; it can be obtained from a Client object.

15.1.2 Constructor & Destructor Documentation

15.1.2.1 ClientEndpoint (Class< TReq > req_class, Class< TRep > rep_class, ClientParams cparams)

Default constructor

15.1.3 Member Function Documentation

15.1.3.1 ClientParams get_client_params ()

Access the [ClientParams](#) defining the configuration of this [ClientEndpoint](#).

15.1.3.2 boolean receive_reply (Sample< TRep > *reply*, SampleIdentity_t *relatedRequestId*) [protected]

Attempts to access a received reply.

This method accepts a parameter of Sample<TRep> & and a SampleIdentity_t. The call will block until a specific reply is found that matches the identity provided the 'relatedRequestId' parameter.

Return values

<i>bool</i>	indicating if a reply was found (true) or not (false).
-------------	--

15.1.3.3 ReturnCode_t send_request (TReq *request*) [protected]

Transmits a request to any matched instances of this service This method accepts a parameter of TReq. Populates request.requestHeader

15.2 ClientParams Class Reference

Public Member Functions

- [ClientParams](#) ()
- [ClientParams](#) ([ClientParams](#) other)
- [ClientParams](#) service_name (String service_name)
- [ClientParams](#) instance_name (String instance_name)
- [ClientParams](#) request_topic_name (String req_topic)
- [ClientParams](#) reply_topic_name (String rep_topic)
- [ClientParams](#) datawriter_qos ([DataWriterQos](#) qos)
- [ClientParams](#) datareader_qos ([DataReaderQos](#) qos)
- [ClientParams](#) publisher ([Publisher](#) publisher)
- [ClientParams](#) subscriber ([Subscriber](#) subscriber)
- [ClientParams](#) domain_participant ([DomainParticipant](#) part)
- String service_name ()
- String instance_name ()
- String request_topic_name ()
- String reply_topic_name ()
- [DataWriterQos](#) datawriter_qos ()
- [DataReaderQos](#) datareader_qos ()
- [Publisher](#) publisher ()
- [Subscriber](#) subscriber ()
- [DomainParticipant](#) domain_participant ()

15.2.1 Detailed Description

Used to pass configuration parameters when constructing a Client.

[ClientParams](#) is a valuetype that serves as a container of configuration parameters of a Client. It is designed to mimic the named-parameters feature available in some programming languages, which improves readability.

15.2.2 Constructor & Destructor Documentation

15.2.2.1 [ClientParams](#) ()

Default constructor

15.2.2.2 [ClientParams](#) ([ClientParams](#) other)

Copy constructor

15.2.3 Member Function Documentation

15.2.3.1 [ClientParams](#) datareader_qos ([DataReaderQos](#) qos)

Assign the [DataReaderQos](#) configured in this instance of [ClientParams](#).

15.2.3.2 **DataReaderQos** `datareader_qos ( )`

Access the [DataReaderQos](#) configured in this instance of [ClientParams](#).

15.2.3.3 **ClientParams** `datawriter_qos ( DataWriterQos qos )`

Assign the [DataWriterQos](#) configured in this instance of [ClientParams](#).

15.2.3.4 **DataWriterQos** `datawriter_qos ( )`

Access the [DataWriterQos](#) configured in this instance of [ClientParams](#).

15.2.3.5 **ClientParams** `domain_participant ( DomainParticipant part )`

Assign the [DomainParticipant](#) configured in this instance of [ClientParams](#).

15.2.3.6 **DomainParticipant** `domain_participant ( )`

Access the [DomainParticipant](#) configured in this instance of [ClientParams](#).

15.2.3.7 **ClientParams** `instance_name ( String instance_name )`

Assign the **instance_name** configured in this instance of [ClientParams](#).

15.2.3.8 **String** `instance_name ( )`

Access the **instance_name** configured in this instance of [ClientParams](#).

15.2.3.9 **ClientParams** `publisher ( Publisher publisher )`

Assign the [Publisher](#) configured in this instance of [ClientParams](#).

15.2.3.10 **Publisher** `publisher ( )`

Access the [Publisher](#) configured in this instance of [ClientParams](#).

15.2.3.11 **ClientParams** `reply_topic_name ( String rep_topic )`

Assign the **reply_topic_name** configured in this instance of [ClientParams](#).

15.2.3.12 **String** `reply_topic_name ( )`

Access the **reply_topic_name** configured in this instance of [ClientParams](#).

15.2.3.13 ClientParams request_topic_name (String *req_topic*)

Assign the **request_topic_name** configured in this instance of [ClientParams](#).

15.2.3.14 String request_topic_name ()

Access the **request_topic_name** configured in this instance of [ClientParams](#).

15.2.3.15 ClientParams service_name (String *service_name*)

Assign the **service_name** configured in this instance of [ClientParams](#).

15.2.3.16 String service_name ()

Access the **service_name** configured in this instance of [ClientParams](#).

15.2.3.17 ClientParams subscriber (Subscriber *subscriber*)

Assign the [Subscriber](#) configured in this instance of [ClientParams](#).

15.2.3.18 Subscriber subscriber ()

Access the [Subscriber](#) configured in this instance of [ClientParams](#).

15.3 ReplierParams Class Reference

Public Member Functions

- [ReplierParams](#) ()
- [ReplierParams](#) ([ReplierParams](#) other)
- [ReplierParams](#) [simple_replier_listener](#) ([SimpleReplierListener](#)<?extends [RequestType](#),?extends [ReplyType](#) > listener)
- [ReplierParams](#) [replier_listener](#) ([ReplierListener](#)<?extends [RequestType](#),?extends [ReplyType](#) > listener)
- [ReplierParams](#) [domain_participant](#) ([DomainParticipant](#) participant)
- [ReplierParams](#) [service_name](#) (String service_name)
- [ReplierParams](#) [instance_name](#) (String instance_name)
- [ReplierParams](#) [request_topic_name](#) (String req_topic)
- [ReplierParams](#) [reply_topic_name](#) (String rep_topic)
- [ReplierParams](#) [datawriter_qos](#) ([DataWriterQos](#) qos)
- [ReplierParams](#) [datareader_qos](#) ([DataReaderQos](#) qos)
- [ReplierParams](#) [publisher](#) ([Publisher](#) publisher)
- [ReplierParams](#) [subscriber](#) ([Subscriber](#) subscriber)
- [DomainParticipant](#) [domain_participant](#) ()
- [ListenerBase](#) [simple_replier_listener](#) ()
- [ListenerBase](#) [replier_listener](#) ()
- String [service_name](#) ()
- String [instance_name](#) ()
- String [request_topic_name](#) ()
- String [reply_topic_name](#) ()
- [DataWriterQos](#) [datawriter_qos](#) ()
- [DataReaderQos](#) [datareader_qos](#) ()
- [Publisher](#) [publisher](#) ()
- [Subscriber](#) [subscriber](#) ()

15.3.1 Detailed Description

Used to pass configuration parameters when constructing a [Replier](#).

[ReplierParams](#) is a valuetype that serves as a container of configuration parameters of a [Replier](#). It is designed to mimic the named-parameters feature available in some programming languages, which improves readability.

15.3.2 Constructor & Destructor Documentation

15.3.2.1 [ReplierParams](#) ()

Default constructor

15.3.2.2 [ReplierParams](#) ([ReplierParams](#) other)

Copy constructor

15.3.3 Member Function Documentation

15.3.3.1 ReplierParams datareader_qos ([DataReaderQos qos](#))

Assign the [DataReaderQos](#) configured in this instance of [ReplierParams](#).

15.3.3.2 DataReaderQos datareader_qos ()

Access the [DataReaderQos](#) configured in this instance of [ReplierParams](#).

15.3.3.3 ReplierParams datawriter_qos ([DataWriterQos qos](#))

Assign the [DataWriterQos](#) configured in this instance of [ReplierParams](#).

15.3.3.4 DataWriterQos datawriter_qos ()

Access the [DataWriterQos](#) configured in this instance of [ReplierParams](#).

15.3.3.5 ReplierParams domain_participant ([DomainParticipant participant](#))

Assign the [DomainParticipant](#) configured in this instance of [ReplierParams](#).

15.3.3.6 DomainParticipant domain_participant ()

Access the [DomainParticipant](#) configured in this instance of [ReplierParams](#).

15.3.3.7 ReplierParams instance_name ([String instance_name](#))

Assign the **instance_name** configured in this instance of [ReplierParams](#).

15.3.3.8 String instance_name ()

Access the **instance_name** configured in this instance of [ReplierParams](#).

15.3.3.9 ReplierParams publisher ([Publisher publisher](#))

Assign the [Publisher](#) configured in this instance of [ReplierParams](#).

15.3.3.10 Publisher publisher ()

Access the [Publisher](#) configured in this instance of [ReplierParams](#).

15.3.3.11 ReplierParams replier_listener ([ReplierListener](#)<?extends RequestType,?extends ReplyType > *listener*)

Assign the [ReplierListener](#) configured in this instance of [ReplierParams](#).

15.3.3.12 ListenerBase replier_listener ()

Access the [ReplierListener](#) configured in this instance of [ReplierParams](#).

15.3.3.13 ReplierParams reply_topic_name (String *rep_topic*)

Assign the **reply_topic_name** configured in this instance of [ReplierParams](#).

15.3.3.14 String reply_topic_name ()

Access the **reply_topic_name** configured in this instance of [ReplierParams](#).

15.3.3.15 ReplierParams request_topic_name (String *req_topic*)

Assign the **request_topic_name** configured in this instance of [ReplierParams](#).

15.3.3.16 String request_topic_name ()

Access the **request_topic_name** configured in this instance of [ReplierParams](#).

15.3.3.17 ReplierParams service_name (String *service_name*)

Assign the **service_name** configured in this instance of [ReplierParams](#).

15.3.3.18 String service_name ()

Access the **service_name** configured in this instance of [ReplierParams](#).

15.3.3.19 ReplierParams simple_replier_listener (SimpleReplierListener<?extends RequestType,?extends ReplyType > *listener*)

Assign the [SimpleReplierListener](#) configured in this instance of [ReplierParams](#).

15.3.3.20 ListenerBase simple_replier_listener ()

Access the [SimpleReplierListener](#) configured in this instance of [ReplierParams](#).

15.3.3.21 ReplierParams subscriber (Subscriber *subscriber*)

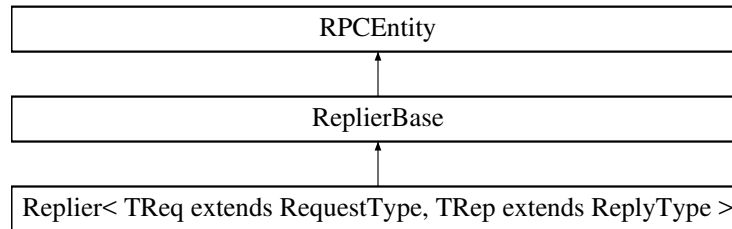
Assign the [Subscriber](#) configured in this instance of [ReplierParams](#).

15.3.3.22 Subscriber subscriber ()

Access the [Subscriber](#) configured in this instance of [ReplierParams](#).

15.4 Replier< TReq extends RequestType, TRep extends ReplyType > Class Template Reference

Inheritance diagram for Replier< TReq extends RequestType, TRep extends ReplyType >:



Public Member Functions

- [Replier](#) (Class< TReq > *treq_clazz*, Class< TRep > *trep_clazz*, [ReplierParams](#) *r_params*)
- void [close](#) ()
- boolean [is_null](#) ()
- void [send_reply](#) (TRep *reply*, SampleIdentity_t *related_request_id*)
- boolean [receive_request](#) (Sample< TReq > *request*, [Duration_t](#) *max_wait*)
- [LoanedSamples](#)< TReq > [receive_requests](#) ([Duration_t](#) *max_wait*)
- [LoanedSamples](#)< TReq > [receive_requests](#) (int *min_request_count*, int *max_request_count*, [Duration_t](#) *max_wait*)
- boolean [wait_for_requests](#) ([Duration_t](#) *max_wait*)
- boolean [wait_for_requests](#) (int *min_count*, [Duration_t](#) *max_wait*)
- boolean [take_request](#) (Sample< TReq > *request*)
- [LoanedSamples](#)< TReq > [take_requests](#) (int *max_samples*)
- boolean [read_request](#) (Sample< TReq > *request*)
- [LoanedSamples](#)< TReq > [read_requests](#) (int *max_samples*)
- [ReplierParams](#) [get_replier_params](#) ()
- boolean [receive_nondata_samples](#) (boolean *enable*)
- [DataReaderI](#)< TReq > [get_request_datareader](#) ()
- [DataWriterI](#)< TRep > [get_reply_datawriter](#) ()

15.4.1 Detailed Description

A replier receives requests and send replies.

An instance of [Replier](#) is configured at the time of construction using [ReplierParams](#), which is a container of configuration parameters such as domain participant, QoS, listeners and more.

[Replier](#) allows listener-based and polling-based reception of requests.

[SimpleReplierListener](#) and [ReplierListener](#) interfaces enable call-back based notification when a request is available. On the other hand, [Replier](#) provides functions to allow polling reception of requests.

15.4.2 Constructor & Destructor Documentation

15.4.2.1 [Replier](#) (Class< TReq > *treq_clazz*, Class< TRep > *trep_clazz*, [ReplierParams](#) *r_params*)

Constructor that accepts [ReplierParams](#) configuration.

15.4.3 Member Function Documentation

15.4.3.1 `void close ( )`

Release the underlying resources including any DDS entities.

The [RPCEntity](#) will no longer be useful and will not participate in any communication.

Implements [RPCEntity](#).

15.4.3.2 `ReplierParams get_replier_params ( )`

Access the [ReplierParams](#) values that define the configuration of the [Replier](#).

15.4.3.3 `DataWriterI<TRep> get_reply_datawriter ( )`

Access the underlying [DataWriter](#) used to send Replies.

15.4.3.4 `DataReaderI<TReq> get_request_datareader ( )`

Access the underlying [DataReader](#) used to receive Requests.

15.4.3.5 `boolean is_null ( )`

Indicates if the entity has been 'closed'.

If `is_null()` is true, then the underlying dds entities have been released and the [RPCEntity](#) is no longer useful for operations.

Implements [RPCEntity](#).

15.4.3.6 `boolean read_request ( Sample< TReq > request )`

Attempt to access the next request.

This may will block until a request is available.

Return values

<i>true</i>	if able to access a request.
<i>false</i>	otherwise

15.4.3.7 `LoanedSamples<TReq> read_requests ( int max_samples )`

Attempt to access up to 'max_samples' requests.

This call may block if no requests are available.

Return values

<i>collection</i>	of requests
-------------------	-------------

15.4.3.8 `boolean receive_nondata_samples ( boolean enable )`

Toggle the underlying entities to deliver non-data samples to the application

Not Yet Supported

15.4.3.9 `boolean receive_request ( Sample< TReq > request, Duration_t max_wait )`

Attempt to access the next request.

This may block for up to 'max_wait' if no request is immediately available.

15.4.3.10 `LoanedSamples<TReq> receive_requests ( Duration_t max_wait )`

Attempt to access any available requests.

This may block for up to 'max_wait' if no requests are immediately available.

15.4.3.11 `LoanedSamples<TReq> receive_requests ( int min_request_count, int max_request_count, Duration_t max_wait )`

Attempt to access up to 'max_request_count' requests.

This may block for up to 'max_wait' if there are not at least 'min_request_count' requests immediately available.

15.4.3.12 `void send_reply ( TRep reply, SampleIdentity_t related_request_id )`

Sends the provided 'reply' associated with 'related_request_id'.

15.4.3.13 `boolean take_request ( Sample< TReq > request )`

Attempt to access the next request.

This call will not block.

Return values

<i>true</i>	if able to access a request.
<i>false</i>	otherwise

15.4.3.14 `LoanedSamples<TReq> take_requests ( int max_samples )`

Attempt to access up to 'max_samples' requests.

This call will not block.

Return values

<i>collection</i>	of requests (possibly empty)
-------------------	------------------------------

15.4.3.15 `boolean wait_for_requests ( Duration_t max_wait )`

Block until at least one request is available.

This may block for up to 'max_wait'.

15.4.3.16 `boolean wait_for_requests ( int min_count, Duration_t max_wait )`

Block until at least 'min_count' requests available.

This may block for up to 'max_wait'.

15.5 RequesterParams Class Reference

Public Member Functions

- [RequesterParams](#) ()
- [RequesterParams](#) ([RequesterParams](#) other)
- [RequesterParams](#) simple_requester_listener ([SimpleRequesterListener](#)<?extends ReplyType > listener)
- [RequesterParams](#) requester_listener ([RequesterListener](#)<?extends RequestType,?extends ReplyType > listener)
- [RequesterParams](#) domain_participant ([DomainParticipant](#) participant)
- [RequesterParams](#) publisher ([Publisher](#) publisher)
- [RequesterParams](#) subscriber ([Subscriber](#) subscriber)
- [RequesterParams](#) datawriter_qos ([DataWriterQos](#) qos)
- [RequesterParams](#) datareader_qos ([DataReaderQos](#) qos)
- [RequesterParams](#) service_name (String name)
- [RequesterParams](#) request_topic_name (String name)
- [RequesterParams](#) reply_topic_name (String name)
- [DomainParticipant](#) domain_participant ()
- [Publisher](#) publisher ()
- [Subscriber](#) subscriber ()
- [DataWriterQos](#) datawriter_qos ()
- [DataReaderQos](#) datareader_qos ()
- [ListenerBase](#) simple_requester_listener ()
- [ListenerBase](#) requester_listener ()
- String service_name ()
- String request_topic_name ()
- String reply_topic_name ()

15.5.1 Detailed Description

Used to pass configuration parameters when constructing a [Requester](#).

[RequesterParams](#) is a valuetype that serves as a container of configuration parameters of a [Requester](#). It is designed to mimic the named-parameters feature available in some programming languages, which improves readability.

15.5.2 Constructor & Destructor Documentation

15.5.2.1 [RequesterParams](#) ()

Default constructor

15.5.2.2 [RequesterParams](#) ([RequesterParams](#) other)

Copy constructor

15.5.3 Member Function Documentation

15.5.3.1 RequesterParams datareader_qos ([DataReaderQos qos](#))

Assign the [DataReaderQos](#) to use.

This QoS will be used when the [Requester](#) creates a [DataReader](#).

15.5.3.2 DataReaderQos datareader_qos ()

Access the [DataReaderQos](#) configured in this instance of [RequesterParams](#).

15.5.3.3 RequesterParams datawriter_qos ([DataWriterQos qos](#))

Assign the [DataWriterQos](#) to use.

This QoS will be used when the [Requester](#) creates a [DataWriter](#).

15.5.3.4 DataWriterQos datawriter_qos ()

Access the [DataWriterQos](#) configured in this instance of [RequesterParams](#).

15.5.3.5 RequesterParams domain_participant ([DomainParticipant participant](#))

Assign the [DomainParticipant](#) to use.

The requester will use this [DomainParticipant](#) to create any required [Publisher](#), [Subscriber](#), [DataWriter](#) and [DataReader](#) entities.

15.5.3.6 DomainParticipant domain_participant ()

Access the [DomainParticipant](#) configured in this instance of [RequesterParams](#).

15.5.3.7 RequesterParams publisher ([Publisher publisher](#))

Assign the [Publisher](#) to use.

The [Requester](#) will use this publisher to create any required [DataWriter](#) entities. If not provided the [Requester](#) will create its own [Publisher](#).

15.5.3.8 Publisher publisher ()

Access the [Publisher](#) configured in this instance of [RequesterParams](#).

15.5.3.9 RequesterParams reply_topic_name ([String name](#))

Assign the 'reply_topic_name'.

This overrides the default topic name generation.

15.5.3.10 String reply_topic_name ()

Access the reply_topic_name configured in this instance of [RequesterParams](#).

15.5.3.11 RequesterParams request_topic_name (String name)

Assign the 'request_topic_name' to use as the Request topic name.

This overrides the default topic name generation.

15.5.3.12 String request_topic_name ()

Access the request_topic_name configured in this instance of [RequesterParams](#).

15.5.3.13 RequesterParams requester_listener (RequesterListener<?extends RequestType,?extends ReplyType > listener)

Assign the 'requester' listener that will be installed in the [Requester](#).

Only one listener may be installed, either the SimpleListener or [RequesterListener](#).

15.5.3.14 ListenerBase requester_listener ()

Access the [RequesterListener](#) configured in this instance of [RequesterParams](#).

15.5.3.15 RequesterParams service_name (String name)

Assign the service_name to use when creating the Request and Reply topic names.

The service_name is used when constructing the topic name used for the Request and Reply topics. If not explicitly defined by the [request_topic_name\(\)](#) configuration item, the request topic is composed of <service_name>_Request. Similarly, if not overridden by [reply_topic_name\(\)](#), the reply topic name is composed of <service_name>_Reply. If the [service_name\(\)](#) item is left unspecified, then the string "Service" is used.

15.5.3.16 String service_name ()

Access the service_name configured in this instance of [RequesterParams](#).

15.5.3.17 RequesterParams simple_requester_listener (SimpleRequesterListener<?extends ReplyType > listener)

Assign the 'simple' listener that will be installed in the [Requester](#).

Only one listener may be installed, either the SimpleListener or [RequesterListener](#).

15.5.3.18 ListenerBase simple_requester_listener ()

Access the [SimpleRequesterListener](#) configured in this instance of [RequesterParams](#).

15.5.3.19 RequesterParams subscriber ([Subscriber](#) *subscriber*)

Assign the [Subscriber](#) to use.

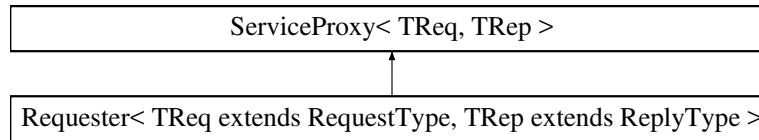
The [Requester](#) will use this subscriber to create any required [DataReader](#) entities. If not provided, the [Requester](#) will create its own [Subscriber](#).

15.5.3.20 [Subscriber](#) subscriber ()

Access the [Subscriber](#) configured in this instance of [RequesterParams](#).

15.6 Requester< TReq extends RequestType, TRep extends ReplyType > Class Template Reference

Inheritance diagram for Requester< TReq extends RequestType, TRep extends ReplyType >:



Public Member Functions

- [Requester](#) (Class< TReq > req_class, Class< TRep > rep_class)
- [Requester](#) (Class< TReq > req_class, Class< TRep > rep_class, [RequesterParams](#) req_params)
- `ReturnCode_t` [send_request](#) (TReq request)
- `ReturnCode_t` [send_request_oneway](#) (TReq request)
- `boolean` [receive_reply](#) (Sample< TRep > reply, [Duration_t](#) timeout)
- `boolean` [receive_reply](#) (Sample< TRep > reply, SampleIdentity_t relatedRequestId)
- [LoanedSamples](#)< TRep > [receive_replies](#) ([Duration_t](#) max_wait)
- [LoanedSamples](#)< TRep > [receive_replies](#) (int min_count, int max_count, [Duration_t](#) max_wait)
- `boolean` [wait_for_replies](#) (int min_count, [Duration_t](#) max_wait)
- `boolean` [wait_for_replies](#) ([Duration_t](#) max_wait)
- `boolean` [wait_for_replies](#) (int min_count, [Duration_t](#) max_wait, SampleIdentity_t related_request_id)
- `boolean` [take_reply](#) (Sample< TRep > reply)
- `boolean` [take_reply](#) (Sample< TRep > reply, SampleIdentity_t related_request_id)
- [LoanedSamples](#)< TRep > [take_replies](#) (int max_count)
- [LoanedSamples](#)< TRep > [take_replies](#) (int max_count, SampleIdentity_t related_request_id)
- [LoanedSamples](#)< TRep > [take_replies](#) (SampleIdentity_t related_request_id)
- `boolean` [read_reply](#) (Sample< TRep > reply)
- `boolean` [read_reply](#) (Sample< TRep > reply, SampleIdentity_t related_request_id)
- [LoanedSamples](#)< TRep > [read_replies](#) (int max_count)
- [LoanedSamples](#)< TRep > [read_replies](#) (int max_count, SampleIdentity_t related_request_id)
- [LoanedSamples](#)< TRep > [read_replies](#) (SampleIdentity_t related_request_id)
- `boolean` [receive_nondata_samples](#) (`boolean` enable)
- [RequesterParams](#) [get_requester_params](#) ()
- `DataWriterI`< TReq > [get_request_datawriter](#) ()
- `DataReaderI`< TRep > [get_reply_datareader](#) ()

15.6.1 Detailed Description

A [Requester](#) sends requests and receives replies.

An instance of a [Requester](#) is configured at the time of construction using [RequesterParams](#), which is a container of configuration parameters, such as domain participant, QoS, listeners and more.

[Requester](#) is inherently asynchronous as sending a request and receiving its corresponding reply (or replies) are separated. [Requester](#) allows listener-based, polling-based, and future-based reception of replies.

[SimpleRequesterListener](#)<TRep> and [RequesterListener](#)<TReq, TRep> interfaces enable callback-based notification when a reply is available. On the other hand, [Requester](#) provides functions to allow polling reception of replies.

Future-based notification of replies is analogous to callback-based notification, however, no request-reply correlation is necessary because every future represents a reply to a unique request.

A requester reference may be bound to a specific service instance. Requests sent through a bound requester reference shall be sent to the bound service instance only.

15.6.2 Constructor & Destructor Documentation

15.6.2.1 Requester (Class< TReq > *treq_class*, Class< TRep > *trep_class*)

Default constructor.

Without any configuration, the [Requester](#) will construct DDS Entities as required in domain '0'.

15.6.2.2 Requester (Class< TReq > *treq_class*, Class< TRep > *trep_class*, RequesterParams *req_params*)

Constructor that accepts [RequesterParams](#) for configuration.

15.6.3 Member Function Documentation

15.6.3.1 DataReaderI<TRep> get_reply_datareader ()

Obtain access to the underlying [DataReader](#) used for receiving Replies.

15.6.3.2 DataWriterI<TReq> get_request_datawriter ()

Obtain access to the underlying [DataWriter](#) used for sending Requests.

15.6.3.3 RequesterParams get_requester_params ()

Obtain the parameters used to configure this Reqeuster.

15.6.3.4 LoanedSamples<TRep> read_replies (int *max_count*)

Attempts to access up to 'max_count' replies.

The available replies (up to 'max_count') are returned in a [LoanedSamples<TRep>](#) object. This routine will not block.

15.6.3.5 LoanedSamples<TRep> read_replies (int *max_count*, SampleIdentity_t *related_request_id*)

Attempts to access up to 'max_count' replies that match 'related_request_id'.

The available matching replies (up to 'max_count') are returned in a [LoanedSamples<TRep>](#) object. This routine will not block.

15.6.3.6 LoanedSamples<TRep> read_replies (SampleIdentity_t *related_request_id*)

Attempts to access all available replies that match 'related_request_id'.

The available matching replies are returned in a [LoanedSamples<TRep>](#) object. This routine will not block.

15.6.3.7 boolean read_reply (Sample< TRep > reply)

Attempts to access the next available reply.

This routine will block until a reply is available. This routines copies the reply data into the provided Sample<TRep> 'reply'.

15.6.3.8 boolean read_reply (Sample< TRep > reply, SampleIdentity_t related_request_id)

Attempts to access the next available reply that matches 'related_reqeust_id'.

This routine will block until a matching reply is available.

15.6.3.9 boolean receive_nondata_samples (boolean enable)

Toggle whether non-data samples are presented to the application.

Note

This is of little utility, as the only non-data samples are unregister, dispose, and not-alive indications which are uncommon with an un-keyed datatype.

15.6.3.10 LoanedSamples<TRep> receive_replies (Duration_t max_wait)

Attempts to access multiple received replies.

This method will access all available replies and return them in a LoanedSamples<TRep> object. If no replies are currently available, the routine will block for up to 'max_wait'.

Return values

<i>boolean</i>	indicating if replies were found (true) or not (false).
----------------	---

15.6.3.11 LoanedSamples<TRep> receive_replies (int min_count, int max_count, Duration_t max_wait)

Attempts to access at least 'min_count' replies.

This method will access all available replies up to 'max_count' and return them in a LoanedSamples<TRep> object. If at least 'min_count' replies are currently not available, the routine will block for up to 'max_wait'.

Return values

<i>boolean</i>	indicating if replies were found (true) or not (false).
----------------	---

15.6.3.12 boolean receive_reply (Sample< TRep > reply, Duration_t timeout)

Attempts to access a received reply.

This method accepts a parameter of Sample<TRep> & and a 'timeout'. The call will block until a reply is found, or the timeout period elapses.

Return values

<i>boolean</i>	indicating if a reply was found (true) or not (false).
----------------	--

15.6.3.13 **boolean** receive_reply (**Sample< TRep > reply**, **SampleIdentity_t relatedRequestId**)

Attempts to access a received reply.

This method accepts a parameter of **Sample<TRep>** & and a **SampleIdentity_t**. The call will block until a specific reply is found that matches the identity provided the 'relatedRequestId' parameter.

Return values

<i>boolean</i>	indicating if a reply was found (true) or not (false).
----------------	--

15.6.3.14 **ReturnCode_t** send_request (**TReq request**)

Transmits a request to any matched instances of this service This method accepts a parameter of **TReq**. Populates request.requestHeader

15.6.3.15 **ReturnCode_t** send_request_oneway (**TReq request**)

Transmits a request to any matched instances of this service This method accepts a parameter of **TReq**.

15.6.3.16 **LoanedSamples<TRep>** take_replies (**int max_count**)

Attempts to access up to 'max_count' replies.

This method will access all available replies up to 'max_count' and return them in a **LoanedSamples<TRep>** object. This routine will not block.

15.6.3.17 **LoanedSamples<TRep>** take_replies (**int max_count**, **SampleIdentity_t related_request_id**)

Attempts to access up to 'max_count' replies that match the 'related_request_id'.

This method will access all available replies up to 'max_count' and return them in a **LoanedSamples<TRep>** object. This routine will not block.

15.6.3.18 **LoanedSamples<TRep>** take_replies (**SampleIdentity_t related_request_id**)

Attempts to access any replies that match the 'related_request_id'.

This method will access all available matching replies and return them in a **LoanedSamples<TRep>** object. This routine will not block.

15.6.3.19 **boolean** take_reply (**Sample< TRep > reply**)

Attempts to access a received reply.

This method accepts a parameter of Sample<TRep> & . The call will not block.

Return values

<i>boolean</i>	indicating if a reply was found (true) or not (false).
----------------	--

15.6.3.20 boolean take_reply (Sample< TRep > reply, SampleIdentity_t related_request_id)

Attempts to access a received reply.

This method accepts a parameter of Sample<TRep> & . The call will not block.

Return values

<i>boolean</i>	indicating if a reply was found (true) or not (false).
----------------	--

15.6.3.21 boolean wait_for_replies (int min_count, Duration_t max_wait)

Blocks until at least 'min_count' replies are available.

If at least 'min_count' replies are currently not available, the routine will block for up to 'max_wait'.

Return values

<i>boolean</i>	indicating if replies were found (true) or not (false).
----------------	---

15.6.3.22 boolean wait_for_replies (Duration_t max_wait)

Blocks until a reply is available.

If no replies are currently available, the routine will block for up to 'max_wait'.

Return values

<i>boolean</i>	indicating if replies were found (true) or not (false).
----------------	---

15.6.3.23 boolean wait_for_replies (int min_count, Duration_t max_wait, SampleIdentity_t related_request_id)

Blocks until at least 'min_count' replies to 'related_request_id' are available.

If at least 'min_count' matching replies are currently not available, the routine will block for up to 'max_wait'.

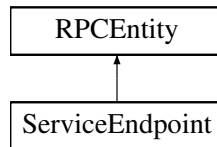
Return values

<i>boolean</i>	indicating if replies were found (true) or not (false).
----------------	---

Not Yet Supported This routine currently ignores 'related_request_id', and will return with success if *any* 'min_count' replies are available.

15.7 ServiceEndpoint Class Reference

Inheritance diagram for ServiceEndpoint:



Public Member Functions

- [ServiceEndpoint](#) ()
- abstract void [close](#) ()
- abstract boolean [is_null](#) ()
- abstract void [pause](#) ()
- abstract void [resume](#) ()
- [ServiceStatus](#) [status](#) ()
- [ServiceParams](#) [get_service_params](#) ()

15.7.1 Detailed Description

A [ServiceEndpoint](#) provides functions to manage a service (e.g., pause, resume, close, etc.) A [ServiceEndpoint](#) shall not be instantiated directly; it can be obtained from a Service object. [ServiceEndpoint](#) is a reference type. All 'Service' implementations derive from [ServiceEndpoint](#).

15.7.2 Constructor & Destructor Documentation

15.7.2.1 ServiceEndpoint ()

Default constructor

15.7.3 Member Function Documentation

15.7.3.1 abstract void close () [abstract]

Release the underlying resources including any DDS entities.

The [RPCEntity](#) will no longer be useful and will not participate in any communication.

Implements [RPCEntity](#).

15.7.3.2 ServiceParams get_service_params ()

Access the [ServiceParams](#) defining the configuration of this [ServiceEndpoint](#).

15.7.3.3 abstract boolean is_null () [abstract]

Indicates if the entity has been 'closed'.

If [is_null\(\)](#) is true, then the underlying dds entities have been released and the [RPCEntity](#) is no longer useful for operations.

Implements [RPCEntity](#).

15.7.3.4 abstract void pause () [abstract]

Pause the operation of the [ServiceEndpoint](#).

This causes the [ServiceEndpoint](#) to not invoke an attached listener.

15.7.3.5 abstract void resume () [abstract]

Resume the operation of the [ServiceEndpoint](#).

This causes the [ServiceEndpoint](#) to invoke any attached listeners upon reception of a request. In order to handle any requests received while the [ServiceEndpoint](#) was paused, the listener will be invoked as part of the [resume\(\)](#) operation.

15.7.3.6 ServiceStatus status ()

Access the current status of the [ServiceEndpoint](#).

15.8 ServiceParams Class Reference

Public Member Functions

- [ServiceParams](#) ()
- [ServiceParams](#) ([ServiceParams](#) other)
- [ServiceParams](#) service_name (String service_name)
- [ServiceParams](#) instance_name (String instance_name)
- [ServiceParams](#) request_topic_name (String req_topic)
- [ServiceParams](#) reply_topic_name (String rep_topic)
- [ServiceParams](#) datawriter_qos ([DataWriterQos](#) qos)
- [ServiceParams](#) datareader_qos ([DataReaderQos](#) qos)
- [ServiceParams](#) publisher ([Publisher](#) publisher)
- [ServiceParams](#) subscriber ([Subscriber](#) subscriber)
- [ServiceParams](#) domain_participant ([DomainParticipant](#) part)
- String service_name ()
- String instance_name ()
- String request_topic_name ()
- String reply_topic_name ()
- [DataWriterQos](#) datawriter_qos ()
- [DataReaderQos](#) datareader_qos ()
- [Publisher](#) publisher ()
- [Subscriber](#) subscriber ()
- [DomainParticipant](#) domain_participant ()

15.8.1 Detailed Description

Used to pass configuration parameters when constructing a Service.

[ServiceParams](#) is a valuetype that serves as a container of configuration parameters of a Service. It is designed to mimic the named-parameters feature available in some programming languages, which improves readability.

15.8.2 Constructor & Destructor Documentation

15.8.2.1 [ServiceParams](#) ()

Default constructor

15.8.2.2 [ServiceParams](#) ([ServiceParams](#) other)

Copy constructor

15.8.3 Member Function Documentation

15.8.3.1 [ServiceParams](#) datareader_qos ([DataReaderQos](#) qos)

Assign the [DataReaderQos](#) configured in this instance of [ServiceParams](#).

15.8.3.2 **DataReaderQos** `datareader_qos ( )`

Access the [DataReaderQos](#) configured in this instance of [ServiceParams](#).

15.8.3.3 **ServiceParams** `datawriter_qos ( DataWriterQos qos )`

Assign the [DataWriterQos](#) configured in this instance of [ServiceParams](#).

15.8.3.4 **DataWriterQos** `datawriter_qos ( )`

Access the [DataWriterQos](#) configured in this instance of [ServiceParams](#).

15.8.3.5 **ServiceParams** `domain_participant ( DomainParticipant part )`

Assign the [DomainParticipant](#) configured in this instance of [ServiceParams](#).

15.8.3.6 **DomainParticipant** `domain_participant ( )`

Access the [DomainParticipant](#) configured in this instance of [ServiceParams](#).

15.8.3.7 **ServiceParams** `instance_name ( String instance_name )`

Assign the **instance_name** configured in this instance of [ServiceParams](#).

15.8.3.8 **String** `instance_name ( )`

Access the **instance_name** configured in this instance of [ServiceParams](#).

15.8.3.9 **ServiceParams** `publisher ( Publisher publisher )`

Assign the [Publisher](#) configured in this instance of [ServiceParams](#).

15.8.3.10 **Publisher** `publisher ( )`

Access the [Publisher](#) configured in this instance of [ServiceParams](#).

15.8.3.11 **ServiceParams** `reply_topic_name ( String rep_topic )`

Assign the **reply_topic_name** configured in this instance of [ServiceParams](#).

15.8.3.12 **String** `reply_topic_name ( )`

Access the **reply_topic_name** configured in this instance of [ServiceParams](#).

15.8.3.13 ServiceParams request_topic_name (String *req_topic*)

Assign the **request_topic_name** configured in this instance of [ServiceParams](#).

15.8.3.14 String request_topic_name ()

Access the **request_topic_name** configured in this instance of [ServiceParams](#).

15.8.3.15 ServiceParams service_name (String *service_name*)

Assign the **service_name** configured in this instance of [ServiceParams](#).

15.8.3.16 String service_name ()

Access the **service_name** configured in this instance of [ServiceParams](#).

15.8.3.17 ServiceParams subscriber (Subscriber *subscriber*)

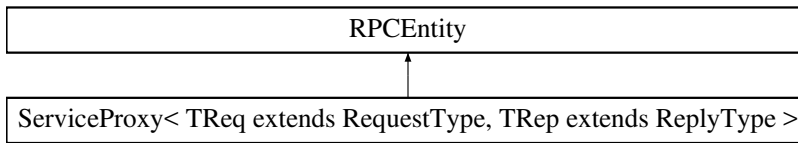
Assign the [Subscriber](#) configured in this instance of [ServiceParams](#).

15.8.3.18 Subscriber subscriber ()

Access the [Subscriber](#) configured in this instance of [ServiceParams](#).

15.9 ServiceProxy< TReq extends RequestType, TRep extends ReplyType > Class Template Reference

Inheritance diagram for ServiceProxy< TReq extends RequestType, TRep extends ReplyType >:



Public Member Functions

- void [close](#) ()
- boolean [is_null](#) ()
- void [bind](#) (String i_name)
- void [unbind](#) ()
- boolean [is_bound](#) ()
- String [get_bound_instance_name](#) ()
- Vector< String > [get_discovered_service_instances](#) ()
- ReturnCode_t [wait_for_service](#) ()
- ReturnCode_t [wait_for_service](#) (Duration_t maxWait)
- ReturnCode_t [wait_for_service](#) (String instanceName)
- ReturnCode_t [wait_for_service](#) (Duration_t max_wait, String instanceName)
- ReturnCode_t [wait_for_services](#) (int count)
- ReturnCode_t [wait_for_services](#) (Duration_t max_wait, int count)
- ReturnCode_t [wait_for_services](#) (Vector< String > instanceNames)
- ReturnCode_t [wait_for_services](#) (Duration_t max_wait, Vector< String > instanceNames)

15.9.1 Detailed Description

ServiceProxy

[ServiceProxy](#) class defines type-independent operations for [Requester](#) and the Client. For example, binding to a specific instance, waiting for an instance to discover, closing the service, and more. [ServiceProxy](#) shall not be instantiated directly.

15.9.2 Member Function Documentation

15.9.2.1 void bind (String i_name)

Bind the [ServiceProxy](#) to service(s) with the provided 'instance_name'.

The [ServiceProxy](#) will communicate only with services that have a matching 'instance_name'.

15.9.2.2 void close ()

Release the underlying resources including any DDS entities.

The [RPCEntity](#) will no longer be useful and will not participate in any communication.

Implements [RPCEntity](#).

15.9.2.3 String get_bound_instance_name ()

Provides the instance_name to which the [ServiceProxy](#) is currently bound.

15.9.2.4 Vector<String> get_discovered_service_instances ()

Access a list of instance_names that have been discovered by the [ServiceProxy](#).

The returned list is a snapshot in time of the currently known services.

15.9.2.5 boolean is_bound ()

Indicates if the [ServiceProxy](#) is currently bound to a particular instance_name.

15.9.2.6 boolean is_null ()

Indicates if the entity has been 'closed'.

If [is_null\(\)](#) is true, then the underlying dds entities have been released and the [RPCEntity](#) is no longer useful for operations.

Implements [RPCEntity](#).

15.9.2.7 void unbind ()

Undo any previous 'bind()' operation.

After this, the [ServiceProxy](#) will communicate with any known services.

15.9.2.8 ReturnCode_t wait_for_service ()

Block until any service is discovered.

Return values

<code>RETCODE_OK</code>	: When matched with requested service[s].
-------------------------	---

15.9.2.9 ReturnCode_t wait_for_service (Duration_t maxWait)

Block up to 'maxWait' until any service is discovered.

Return values

<i>RETCODE_OK</i>	: If matched with requested service[s].
<i>RETCODE_TIMEOUT</i>	: If the maxWait time passes without discovering a service.

15.9.2.10 `ReturnCode_t wait_for_service ( String instanceName )`

Wait for a service with matching 'instanceName' to be discovered.

Return values

<i>RETCODE_OK</i>	: When matched with requested service[s].
-------------------	---

15.9.2.11 `ReturnCode_t wait_for_service ( Duration_t max_wait, String instanceName )`

Block up to 'maxWait' until a service matching 'instanceName' is discovered.

Return values

<i>RETCODE_OK</i>	: If matched with requested service[s].
<i>RETCODE_TIMEOUT</i>	: If the maxWait time passes without discovering a service.

15.9.2.12 `ReturnCode_t wait_for_services ( int count )`

Block until at least 'count' services have been discovered.

Return values

<i>RETCODE_OK</i>	: When matched with requested service[s].
-------------------	---

15.9.2.13 `ReturnCode_t wait_for_services ( Duration_t max_wait, int count )`

Block until at least 'count' services have been discovered or 'maxWait' time elapses.

Return values

<i>RETCODE_OK</i>	: If matched with requested service[s].
<i>RETCODE_TIMEOUT</i>	: If the maxWait time passes without discovering 'count' services.

15.9.2.14 `ReturnCode_t wait_for_services ( Vector< String > instanceNames )`

Block until discovering a service for each of the listed 'instanceNames'.

Return values

<i>RETCODE_OK</i>	: When matched with requested service[s].
-------------------	---

15.9.2.15 `ReturnCode_t wait_for_services ( Duration_t max_wait, Vector< String > instanceNames )`

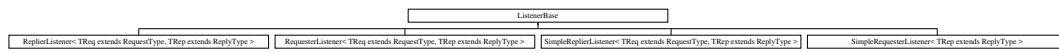
Block until discovering a service for each of the listed 'instanceNames', or 'maxWait' elapses.

Return values

<i>RETCODE_OK</i>	: If matched with requested service[s].
<i>RETCODE_TIMEOUT</i>	: If the maxWait time passes without discovering the specified services.

15.10 ListenerBase Interface Reference

Inheritance diagram for ListenerBase:

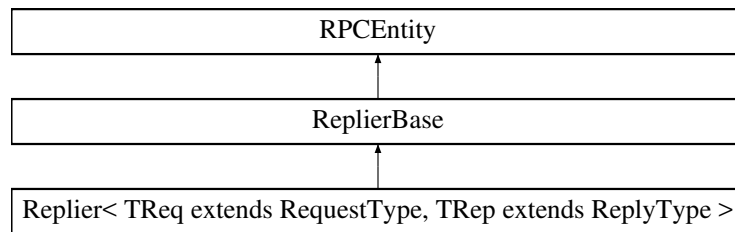


15.10.1 Detailed Description

Basis for all rpc related listener classes.

15.11 ReplierBase Interface Reference

Inheritance diagram for ReplierBase:



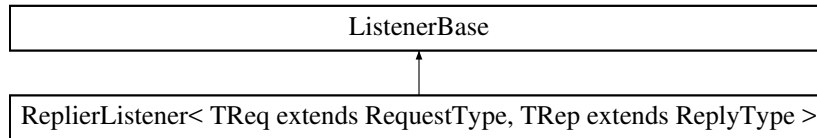
Additional Inherited Members

15.11.1 Detailed Description

Provides a generic basis for all [Replier](#) objects.

15.12 ReplierListener< TReq extends RequestType, TRep extends ReplyType > Interface Template Reference

Inheritance diagram for ReplierListener< TReq extends RequestType, TRep extends ReplyType >:



Public Member Functions

- void [on_request_available](#) ([Replier](#)< TReq, TRep > replier)

15.12.1 Detailed Description

ReplierListener can be installed on a [Replier](#).

[ReplierListener](#)<TReq, TRep> interface is used to provide an asynchronous request listener for a [Replier](#). It is passed to the [Replier](#) constructor through [ReplierParams](#). It extends [ListenerBase](#) interface and enables asynchronous processing of the requests. The callback is provided the [Replier](#) object and returns void.

See also

[ReplierParams](#)

15.12.2 Member Function Documentation

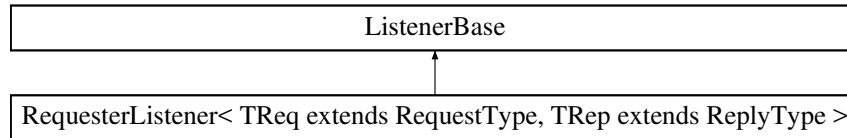
15.12.2.1 void on_request_available ([Replier](#)< TReq, TRep > *replier*)

Invoked when the [Replier](#) has one or more requests pending.

The method can process the available requests and send replies as required.

15.13 RequesterListener< TReq extends RequestType, TRep extends ReplyType > Interface Template Reference

Inheritance diagram for RequesterListener< TReq extends RequestType, TRep extends ReplyType >:



Public Member Functions

- void [on_reply_available](#) ([Requester](#)< TReq, TRep > requester)

15.13.1 Detailed Description

[RequesterListener](#) can be installed on a [Requester](#).

[RequesterListener](#)<TReq,TRep> interface is used to receive notification of reply arrival. It is passed to the [Requester](#) through [RequesterParams](#). It extends [ListenerBase](#) and enables processing of replies. The callback is provided the replier object and returns void.

See also

[RequesterParams](#)

15.13.2 Member Function Documentation

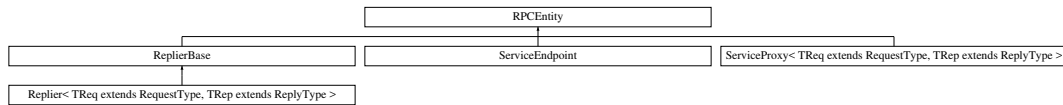
15.13.2.1 void on_reply_available ([Requester](#)< TReq, TRep > *requester*)

Invoked when one or more replies are available.

Can process the replies as appropriate.

15.14 RPCEntity Interface Reference

Inheritance diagram for RPCEntity:



Public Member Functions

- void [close](#) ()
- boolean [is_null](#) ()

15.14.1 Detailed Description

[RPCEntity](#) is a base abstract class.

[RPCEntity](#) is the base abstract class extended by all the active RPC entities. It supports [close\(\)](#) and [is_null\(\)](#) operations.

15.14.2 Member Function Documentation

15.14.2.1 void [close](#) ()

Release the underlying resources including any DDS entities.

The [RPCEntity](#) will no longer be useful and will not participate in any communication.

Implemented in [Replier< TReq extends RequestType, TRep extends ReplyType >](#), [ServiceProxy< TReq extends RequestType, TRep extends ReplyType >](#), and [ServiceEndpoint](#).

15.14.2.2 boolean [is_null](#) ()

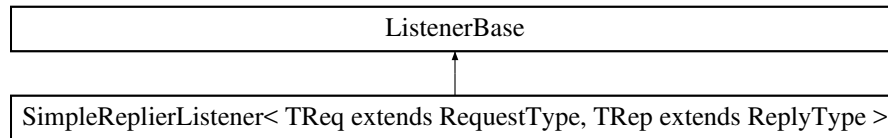
Indicates if the entity has been 'closed'.

If [is_null\(\)](#) is true, then the underlying dds entities have been released and the [RPCEntity](#) is no longer useful for operations.

Implemented in [Replier< TReq extends RequestType, TRep extends ReplyType >](#), [ServiceProxy< TReq extends RequestType, TRep extends ReplyType >](#), and [ServiceEndpoint](#).

15.15 SimpleReplierListener< TReq extends RequestType, TRep extends ReplyType > Interface Template Reference

Inheritance diagram for SimpleReplierListener< TReq extends RequestType, TRep extends ReplyType >:



Public Member Functions

- TRep [process_request](#) (Sample< TReq > sample, SampleIdentity_t sample_id)

15.15.1 Detailed Description

[SimpleReplierListener](#) can be installed on a [Replier](#).

[SimpleReplierListener<TReq, TRep>](#) is used to provide a synchronous request listener for a [Replier](#). [ReplierParams](#) is used to pass an instance of [SimpleReplierListener<TReq, TRep>](#). It extends the [ListenerBase](#) abstract class and enables synchronous processing of the requests. The callback is provided the request object and must return a reply.

See also

[ReplierParams](#)

15.15.2 Member Function Documentation

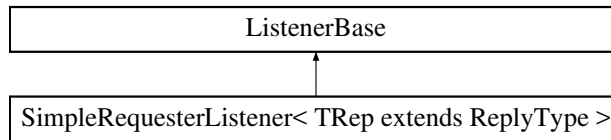
15.15.2.1 TRep process_request (Sample< TReq > *sample*, SampleIdentity_t *sample_id*)

Invoked when the [Replier](#) has recieved a request.

Must initilaize and return a Reply.

15.16 SimpleRequesterListener< TRep extends ReplyType > Interface Template Reference

Inheritance diagram for SimpleRequesterListener< TRep extends ReplyType >:



Public Member Functions

- void [process_reply](#) (Sample< TRep > sample, SampleIdentity_t sample_identity)

15.16.1 Detailed Description

[SimpleRequesterListener](#) can be installed on a [Requester](#).

[SimpleRequesterListener<TRep>](#) abstract class is used to receive notifications of arrival of replies. It is passed to the [Requester](#) through [RequesterParams](#). It extends [ListenerBase](#) and enables processing of replies. The callback is provided access to the reply sample and returns void.

See also

[RequesterParams](#)

15.16.2 Member Function Documentation

15.16.2.1 void [process_reply](#) (Sample< TRep > *sample*, SampleIdentity_t *sample_identity*)

Invoked when a reply is available.

Must process the reply.

15.17 ServiceStatus Enum Reference

Public Attributes

- [CLOSED](#)
- [PAUSED](#)
- [RUNNING](#)

15.17.1 Detailed Description

Indicates the status of a [ServiceEndpoint](#)

15.17.2 Member Data Documentation

15.17.2.1 CLOSED

The [ServiceEndpoint](#) has been closed.

15.17.2.2 PAUSED

The [ServiceEndpoint](#) is paused.

15.17.2.3 RUNNING

The [ServiceEndpoint](#) is running.

Chapter 16

CoreDX DDS DynamicTypes

NOTE: This API is deprecated. The preferred DynamicType API is the one provided by the XTypes implementation.

Chapter 17

Data Structure Index

17.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

ClientEndpoint< TReq extends RequestType, TRep extends ReplyType >	229
ClientParams	231
Condition A Condition can be added to a DDS_WaitSet to provide synchronous event notification	21
Constants Provides various constants and utility routines used throughout CoreDX DDS	191
ContentFilteredTopic ContentFilteredTopic provides a topic that may include data filtered from a related Topic	22
DataReader The DataReader entity allows the application to subscribe to and read data	24
DataReaderListener The DataReaderListener provides asynchronous notification of DataReader events. This listener can be installed during DataReader creation, DomainParticipant_create_yyy(), as well as by calling DataReader_set_listener()	153
DataReaderQos Structure that holds DataReader Quality of Service policies	89
DataRepresentationQosPolicy This QoS policy establishes the DataRepresentation (data encoding) offered by a DataWriter or requested by a DataReader	106
DataWriter The DataWriter entity provides an interface for the application to publish (write) data. The DataWriter is an abstract class that is extended to support a particular data type required by the application. A DataReader is associated with, and writes on, a single Topic	29
DataWriterListener The DataWriterListener provides asynchronous notification of DataWriter events. This listener can be installed during DataWriter creation, Publisher_create_datawriter(), as well as by calling DataWriter_set_listener()	155
DataWriterQos Structure that holds DataWriter Quality of Service policies	92
DeadlineQosPolicy This QoS policy establishes a minimum update period for data instances	105
DestinationOrderQosPolicy This QoS policy controls how each Subscriber orders received data samples	107

DestinationOrderQosPolicyKind	
This enumeration contains the kinds of Destination Ordering	139
DiscoveryQosPolicy	
QoS Policy for configuring aspects of the Discovery and Builtin entities	108
DiscoveryQosPolicyDiscoveryKind	
This enumeration contains the kinds of Discovery	140
DomainEntity	
Base class for all DDS Domain Entities	33
DomainId_t	179
DomainParticipant	
The DomainParticipant is used to configure, create and destroy Publisher , Subscriber and Topic objects	37
DomainParticipantFactory	
DomainParticipantFactory constructs DomainParticipants. The	34
DomainParticipantFactoryQos	
Structure that holds DomainParticipantFactory Quality of Service policies	95
DomainParticipantListener	157
DomainParticipantQos	
Structure that holds DomainParticipant Quality of Service policies	96
DurabilityQosPolicy	109
DurabilityQosPolicyKind	
This enumeration contains the kinds of Durability	141
DurabilityServiceQosPolicy	110
Duration_t	
Represents a time duration with nanosecond resolution	180
Entity	
Base class for all DDS Entities	47
EntityFactoryQosPolicy	111
EntityNameQosPolicy	112
FooDataReader	
The FooDataReader is an example specialized DataReader (generated by the <code>coredx_ddl</code> compiler) for reading data samples of type 'Foo'	77
FooDataWriter	
The FooDataWriter is an example specialized DataWriter (generated by the <code>coredx_ddl</code> compiler) for writing data samples of type 'Foo'	84
FooTypeSupport	
Implements the TypeSupport interface for the Data Type 'Foo'	87
GroupDataQosPolicy	
Allows the application to attach arbitrary information to a Publisher or Subscriber	113
GuardCondition	
A GuardCondition is a Condition where the trigger_value is under application control	48
HistoryQosPolicy	
Controls the amount of historical data maintained by a DataReader or DataWriter	114
HistoryQosPolicyKind	
This enumeration contains the kinds of History	142
InconsistentTopicStatus	
InconsistentTopicStatus provides status related to the <code>on_inconsistent_topic</code> listener methods of the TopicListener structure	165
InstanceHandle_t	
Used to identify a DDS Entity or data instance. An InstanceHandle is unique within its context. For example, each data instance known by a DataReader is assigned a unique InstanceHandle. However, these handles may not be unique when compared to handles from another DataReader	181
IpTransportInterface	
Defines an interface for use by the UdpTransport	215

LatencyBudgetQosPolicy	
Specifies allowable latency	115
LifespanQosPolicy	
Specifies the maximum duration of validity of the data written by the DataWriter	116
ListenerBase	260
LivelinessChangedStatus	
Status related to the on_liveliness_changed listener methods of the DataReader , Subscriber , and DomainParticipant structures	166
LivelinessLostStatus	
Status related to the on_liveliness_lost listener methods of the DataWriter , Publisher , and DomainParticipant structures	167
LivelinessQosPolicy	
Determines the mechanism and parameters used by the application to determine whether an Entity is alive	117
LivelinessQosPolicyKind	
This enumeration contains the kinds of Liveliness	143
LmtTransport	
An instance of a LMT Transport that can be added to a DomainParticipant	218
LmtTransportConfig	
Structure that holds LMT Transport configuration items	216
LoanedSamples< T >	177
Locator	
Network address	49
LoggingQosPolicy	
Controls the amount and kind of data that is logged	118
MultiTopic	
MultiTopic provides a topic that may include data from multiple Topics	50
OfferedDeadlineMissedStatus	
Status related to the on_offered_deadline_missed listener methods of the DataWriter , Publisher , and DomainParticipant structures	168
OfferedIncompatibleQosStatus	
Status related to the on_offered_incompatible_qos listener methods of the DataWriter , Publisher , and DomainParticipant structures	169
OwnershipQosPolicy	
Determines instance ownership in the case of multiple writers. CoreDX DDS supports both SHARED_OWNERSHIP_QOS and EXCLUSIVE_OWNERSHIP_QOS	119
OwnershipQosPolicyKind	
This enumeration contains the kinds of Ownership	144
OwnershipStrengthQosPolicy	
Defines the strength, or priority, of a Writer. The strength is used to determine ownership in the case of EXCLUSIVE_OWNERSHIP_QOS. When multiple writers publish data about the same instance, the stronger writer is considered the owner, and data from other writers is not delivered to the reader 120	
ParticipantBuiltinTopicData	
The ParticipantBuiltinTopicData is a built-in Data Type that represents Participant Discovery information	186
ParticipantBuiltinTopicDataDataReader	
The ParticipantBuiltinTopicDataDataReader is a built-in DataReader that can be used to access Participant Discovery information. The reader can be accessed through the special 'builtin' Subscriber via the DomainParticipant.get_builtin_subscriber() routine	185
ParticipantLocator	
Describes a the location and identity of a potential peer DomainParticipant	51
PartitionQosPolicy	
Defines a logical data partition	121

PeerParticipantQosPolicy	
Configures a list of DomainParticipant peers to attempt communication with	122
PresentationQosPolicy	
Controls the presentation of received data samples to the application. CoreDX DDS currently supports only the access_scope = INSTANCE_PRESENTATION_QOS policy	123
PresentationQosPolicyAccessScopeKind	
This enumeration contains the kinds of Access Scope for the PresentationQosPolicy	145
Property_t	
A name-value pair property. The 'propagate' flag indicates if this property should be transferred through discovery	182
PropertyQosPolicy	
The PropertyQosPolicy supports adding arbitrary name-value pairs	124
PublicationBuiltinTopicData	
The PublicationBuiltinTopicData is a built-in Data Type that represents Publication Discovery information	188
PublicationBuiltinTopicDataDataReader	
The PublicationBuiltinTopicDataDataReader is a built-in DataReader that can be used to access Publication Discovery information. The reader can be accessed through the special 'builtin' Subscriber via the DomainParticipant.get_builtin_subscriber() routine	187
PublicationMatchedStatus	
Status related to the on_publication_matched listener methods of the DataWriter , Publisher , and DomainParticipant structures	170
Publisher	
The Publisher configures, creates, manages and destroys DataWriters	52
PublisherListener	160
PublisherQos	
Structure that holds Publisher Quality of Service policies	98
QosPolicyCount	
A QoS Policy identifier, along with a count	56
QosProvider	
QosProvider loads QoS settings from a library and provides interfaces to access entity specific QoS Policies	151
QueryCondition	
The trigger_value is driven by the data available, after applying the filter, in the associated DataReader	57
ReadCondition	
A ReadCondition is a specialized Condition associated with a DataReader	59
ReaderDataLifecycleQosPolicy	
Specifies the lifecycle behavior of data instances managed by the DataReader	125
ReliabilityQosPolicy	
Indicates the level of reliability offered/provided by the Entity . If kind is RELIABLE_RELIABILITY_QOS, then the middleware will attempt to deliver all samples in the history cache. If samples are not received, then they will be retried	126
ReliabilityQosPolicyKind	
This enumeration contains the kinds of Reliability	146
Replier< TReq extends RequestType, TRep extends ReplyType >	237
ReplierBase	261
ReplierListener< TReq extends RequestType, TRep extends ReplyType >	262
ReplierParams	234
RequestedDeadlineMissedStatus	
Status related to the on_requested_deadline_missed listener methods of the DataReader , Subscriber , and DomainParticipant structures	171
RequestedIncompatibleQosStatus	172
Requester< TReq extends RequestType, TRep extends ReplyType >	245

RequesterListener< TReq extends RequestType, TRep extends ReplyType >	263
RequesterParams	241
ResourceLimitsQosPolicy	
Specifies the resources that the Service can use to maintain data samples and instances	127
RPCEntity	264
RpcQosPolicy	128
RTPSReaderQosPolicy	
QoS Policy for configuring aspects of the RTPS Reader Protocol	129
RTPSWriterQosPolicy	
QoS Policy for configuring aspects of the RTPS Writer Protocol	130
SampleInfo	
The SampleInfo structure contains information associated with each Sample. The DataReader.read() and take() operations return two vectors. One vector contains Sample(s) and the other contains SampleInfo(s) . There is a one-to-one correspondence between items in these two vectors. Each Sample is described by the corresponding SampleInfo instance	61
SampleLostStatus	
Status related to the on_sample_lost listener methods of the DataReader , Subscriber , and DomainParticipant structures	173
SampleRejectedStatus	
Status related to the on_sample_rejected listener methods of the DataReader , Subscriber , and DomainParticipant structures	174
ServiceEndpoint	251
ServiceParams	253
ServiceProxy< TReq extends RequestType, TRep extends ReplyType >	256
ServiceStatus	267
SimpleReplierListener< TReq extends RequestType, TRep extends ReplyType >	265
SimpleRequesterListener< TRep extends ReplyType >	266
SslTransport	
An instance of a SSL Transport that can be added to a DomainParticipant	??
SslTransportConfig	
Structure that holds SSL Transport configuration items	??
StatusCondition	
A StatusCondition is a condition associated with an Entity . The trigger_value is driven by the communication status of the associated Entity	64
Subscriber	
The Subscriber configures, creates, manages and destroys DataReaders	65
SubscriberListener	
The SubscriberListener provides asynchronous notification of Subscriber events. This listener can be installed during Subscriber creation, DomainParticipant.create_subscriber() as well as by calling Subscriber.set_listener()	162
SubscriberQos	
Structure that holds DDS_Subscriber Quality of Service policies	100
SubscriptionBuiltinTopicData	
The SubscriptionBuiltinTopicData is a built-in Data Type that represents Subscription Discovery information	190
SubscriptionBuiltinTopicDataReader	
The SubscriptionBuiltinTopicDataReader is a built-in DataReader that can be used to access Subscription Discovery information	189
SubscriptionMatchedException	
Status related to the on_subscription_matched listener methods of the DataReader , Subscriber , and DomainParticipant structures	175
TcpTransport	
An instance of a TCP Transport that can be added to a DomainParticipant	221

TcpTransportConfig	Structure that holds TCP Transport configuration items	219
ThreadModelQosPolicy	QoS Policy for configuring the threading behavior of the DomainParticipant	131
Time_t	Represents time with nanosecond resolution	183
TimeBasedFilterQosPolicy	Defines a filter based on time between samples. The DataReader indicates that it wants at most one sample for each instance every minimum_separation interval	132
Topic	Topic is the basic description of data to be published or subscribed. A topic is identified by a name and a type . A Topic is created by calling DomainParticipant.create_topic() . Prior to creating a Topic , the associated data type must be registered with the DomainParticipant via a call to the TypeSupport-XYZ.register_type() function. [The register_type() function is auto-generated 'type-specific' code.] . .	69
TopicDataQosPolicy	Allows the application to attach arbitrary information to a Topic QoS	133
TopicDescription	TopicDescription is an abstract that provides the foundation for Topic , ContentFilteredTopic , and MultiTopic	71
TopicListener	The DDS_TopicListener provides asynchronous notification of DDS_Topic events. This listener can be installed during Topic creation (DDS_DomainParticipant_create_topic() and related) as well as by calling DDS_Topic_set_listener()	164
TopicQos	Structure that holds DDS_Topic Quality of Service policies	102
Transport	An instance of a Transport that can be added to a DomainParticipant	222
TransportPriorityQosPolicy	A hint to the middleware to help configure the transport priority mechanism	134
TypeCodeKind	This enumeration contains the kinds of data types used in constructing a Typecode	147
TypecodeQosPolicy	Typecode representing the datatype a DataReader reads or a DataWriter writes	135
TypeConsistencyEnforcementQosPolicy	This QoS policy establishes the policy for establishing consistency of the Reader and Writer data types	136
TypeConsistencyKind	This enumeration contains the kinds of Reliability	150
TypeSupport		72
UdpTransport	An instance of a Transport that can be added to a DomainParticipant	227
UdpTransportConfig	Structure that holds UDP Transport configuration items	223
UserDataQosPolicy	Allows the application to attach arbitrary information to a DomainParticipant , DataWriter or DataReader	137
WaitSet	A DDS_WaitSet maintains a set of Condition objects and allows the application to wait until one or more of them have a trigger_value of TRUE	74
WriterDataLifecycleQosPolicy	Specifies the lifecycle behavior of data instances managed by the DataWriter . If autodispose_unregistered_instances is true, then the DataWriter will automatically dispose any instances that are unregistered. Note: When a DataWriter is deleted, it will automatically unregister all of its instances. With this policy == true, then all instances will also be disposed	138

[SslTransport.X509_Der](#) ??

Chapter 18

Not Yet Supported

Member [DataReader.create_querycondition](#) (long sample_states, long view_states, long instance_states, String query_expression, Vector query_parameters)

QueryConditions are not yet supported as triggers for a [WaitSet](#).

Member [DomainParticipant.create_multitopic](#) (String name, String type_name, String subscription_expression, Vector expression_params)

This is currently unsupported in the Java language binding.

Member [DomainParticipant.delete_multitopic](#) ([MultiTopic](#) a_multitopic)

This is currently unsupported in the Java language binding.

Member [DomainParticipant.get_discovered_topic_data](#) ([TopicBuiltinTopicData](#) topic_data, [InstanceHandle_t](#) topic_handle)

This is currently unsupported in the Java language binding.

Member [DomainParticipant.get_discovered_topics](#) (Vector topic_handles)

This is currently unsupported in the Java language binding.

Member [DomainParticipant.ignore_participant](#) ([InstanceHandle_t](#) handle)

This is currently unsupported in the Java language binding.

Member [DomainParticipant.ignore_publication](#) ([InstanceHandle_t](#) handle)

This is currently unsupported in the Java language binding.

Member [DomainParticipant.ignore_subscription](#) ([InstanceHandle_t](#) handle)

This is currently unsupported in the Java language binding.

Member [DomainParticipant.ignore_topic](#) ([InstanceHandle_t](#) handle)

This is currently unsupported in the Java language binding.

Class [MultiTopic](#)

This is currently unsupported in the Java language binding.

Member [MultiTopic.get_name](#) ()

This routine is not yet implemented.

Member [MultiTopic.get_participant](#) ()

This routine is not yet implemented.

Member [MultiTopic.get_type_name](#) ()

This routine is not yet implemented.

Member [Publisher.begin_coherent_changes](#) ()

This operation is not yet implemented.

Member [Publisher.end_coherent_changes](#) ()

This operation is not yet implemented.

Member [Publisher.resume_publications](#) ()

This operation is not yet implemented.

Member [Publisher.suspend_publications](#) ()

This operation is not yet implemented.

Class [QueryCondition](#)

CoreDX DDS does not yet support QueryConditions as triggers for a [WaitSet](#).

Member [Replier< TReq extends RequestType, TRep extends ReplyType >.receive_nondata_samples](#) (boolean enable)**Member [Requester< TReq extends RequestType, TRep extends ReplyType >.wait_for_replies](#) (int min_count, [Duration_t](#) max_wait, [SampleIdentity_t](#) related_request_id)**

This routine currently ignores 'related_request_id', and will return with success if *any* 'min_count' replies are available.

Member [Subscriber.begin_access](#) ()

This is currently unsupported in the Java language binding.

Member [Subscriber.end_access](#) ()

This is currently unsupported in the Java language binding.

Member [Subscriber.get_datareaders](#) (Vector readers, long sample_states, long view_states, long instance_states)

This is currently unsupported in the Java language binding.

Chapter 19

Deprecated List

Member [DomainParticipant.add_tcp_transport \(\)](#)

Chapter 20

Hierarchical Index

20.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

ClientParams	231
Condition	21
GuardCondition	48
ReadCondition	59
QueryCondition	57
StatusCondition	64
Constants	191
DataReaderListener	153
DataReaderQos	89
DataRepresentationQosPolicy	106
DataWriterListener	155
DataWriterQos	92
DeadlineQosPolicy	105
DestinationOrderQosPolicy	107
DestinationOrderQosPolicyKind	139
DiscoveryQosPolicy	108
DiscoveryQosPolicyDiscoveryKind	140
DomainId_t	179
DomainParticipantFactory	34
DomainParticipantFactoryQos	95
DomainParticipantListener	157
DomainParticipantQos	96
DurabilityQosPolicy	109
DurabilityQosPolicyKind	141
DurabilityServiceQosPolicy	110
Duration_t	180
Entity	47
DomainEntity	33
DataReader	24
FooDataReader	77
ParticipantBuiltinTopicDataDataReader	185
PublicationBuiltinTopicDataDataReader	187
SubscriptionBuiltinTopicDataDataReader	189

DataWriter	29
FooDataWriter	84
Publisher	52
Subscriber	65
Topic	69
DomainParticipant	37
EntityFactoryQosPolicy	111
EntityNameQosPolicy	112
GroupDataQosPolicy	113
HistoryQosPolicy	114
HistoryQosPolicyKind	142
InconsistentTopicStatus	165
InstanceHandle_t	181
IpTransportInterface	215
LatencyBudgetQosPolicy	115
LifespanQosPolicy	116
ListenerBase	260
ReplierListener< TReq extends RequestType, TRep extends ReplyType >	262
RequesterListener< TReq extends RequestType, TRep extends ReplyType >	263
SimpleReplierListener< TReq extends RequestType, TRep extends ReplyType >	265
SimpleRequesterListener< TRep extends ReplyType >	266
LivelinessChangedStatus	166
LivelinessLostStatus	167
LivelinessQosPolicy	117
LivelinessQosPolicyKind	143
LmtTransportConfig	216
LoanedSamples< T >	177
Locator	49
LoggingQosPolicy	118
OfferedDeadlineMissedStatus	168
OfferedIncompatibleQosStatus	169
OwnershipQosPolicy	119
OwnershipQosPolicyKind	144
OwnershipStrengthQosPolicy	120
ParticipantBuiltinTopicData	186
ParticipantLocator	51
PartitionQosPolicy	121
PeerParticipantQosPolicy	122
PresentationQosPolicy	123
PresentationQosPolicyAccessScopeKind	145
Property_t	182
PropertyQosPolicy	124
PublicationBuiltinTopicData	188
PublicationMatchedStatus	170
PublisherListener	160
PublisherQos	98
QosPolicyCount	56
QosProvider	151
ReaderDataLifecycleQosPolicy	125
ReliabilityQosPolicy	126
ReliabilityQosPolicyKind	146
Replier< TReq, TRep >	237
ReplierListener< TReq, TRep >	262
ReplierParams	234

RequestedDeadlineMissedStatus	171
RequestedIncompatibleQosStatus	172
RequesterListener< TReq, TRep >	263
RequesterParams	241
ResourceLimitsQosPolicy	127
RPCEntity	264
ReplierBase	261
Replier< TReq extends RequestType, TRep extends ReplyType >	237
ServiceEndpoint	251
ServiceProxy< TReq extends RequestType, TRep extends ReplyType >	256
RpcQosPolicy	128
RTPSReaderQosPolicy	129
RTPSWriterQosPolicy	130
SampleInfo	61
SampleLostStatus	173
SampleRejectedStatus	174
ServiceParams	253
ServiceProxy< TReq, TRep >	256
ClientEndpoint< TReq extends RequestType, TRep extends ReplyType >	229
Requester< TReq extends RequestType, TRep extends ReplyType >	245
ServiceStatus	267
SimpleReplierListener< TReq, TRep >	265
SimpleRequesterListener< TRep >	266
SslTransportConfig	??
SubscriberListener	162
SubscriberQos	100
SubscriptionBuiltinTopicData	190
SubscriptionMatchedStatus	175
TcpTransportConfig	219
ThreadModelQosPolicy	131
Time_t	183
TimeBasedFilterQosPolicy	132
TopicDataQosPolicy	133
TopicDescription	71
ContentFilteredTopic	22
MultiTopic	50
Topic	69
TopicListener	164
TopicQos	102
Transport	222
LmtTransport	218
SslTransport	??
TcpTransport	221
UdpTransport	227
TransportPriorityQosPolicy	134
TypeCodeKind	147
TypecodeQosPolicy	135
TypeConsistencyEnforcementQosPolicy	136
TypeConsistencyKind	150
TypeSupport	72
FooTypeSupport	87
UdpTransportConfig	223
UserDataQosPolicy	137

WaitSet	74
WriterDataLifecycleQosPolicy	138
SslTransport.X509_Der	??

Chapter 21

Namespace Details

21.1 Packages

Here are the packages with brief descriptions (if available):

[com.toc.coredx.DDS.rpc](#) 287

21.2 Package `com.toc.coredx.DDS.rpc`

Classes

- class [ClientEndpoint](#)
- class [ClientParams](#)
- interface [ListenerBase](#)
- class [Replier](#)
- interface [ReplierBase](#)
- interface [ReplierListener](#)
- class [ReplierParams](#)
- class [Requester](#)
- interface [RequesterListener](#)
- class [RequesterParams](#)
- interface [RPCEntity](#)
- class [ServiceEndpoint](#)
- class [ServiceParams](#)
- class [ServiceProxy](#)
- enum [ServiceStatus](#)
- interface [SimpleReplierListener](#)
- interface [SimpleRequesterListener](#)

21.2.1 Detailed Description

Provides the RPC over DDS infrastructure

Index

- ALIVE_INSTANCE_STATE
 - DDS Conditions, Listeners, and WaitSets, [10](#)
- ALL_STATUS
 - DDS Status Structures, [15](#)
- ALLOW_TYPE_COERCION
 - com::toc::coredx::DDS::TypeConsistencyKind, [150](#)
- ANY_INSTANCE_STATE
 - DDS Conditions, Listeners, and WaitSets, [10](#)
- ANY_SAMPLE_STATE
 - DDS Conditions, Listeners, and WaitSets, [10](#)
- ANY_VIEW_STATE
 - DDS Conditions, Listeners, and WaitSets, [10](#)
- AUTOMATIC_LIVELINESS_QOS
 - com::toc::coredx::DDS::LivelinessQosPolicyKind, [143](#)
- absolute_generation_rank
 - com::toc::coredx::DDS::SampleInfo, [61](#)
- add_tcp_transport
 - com::toc::coredx::DDS::DomainParticipant, [38](#)
- add_transport
 - com::toc::coredx::DDS::DomainParticipant, [38](#)
- addr
 - com::toc::coredx::DDS::Locator, [49](#)
- advertise_meta_multicast
 - com::toc::coredx::DDS::UdpTransportConfig, [224](#)
- advertise_user_multicast
 - com::toc::coredx::DDS::UdpTransportConfig, [224](#)
- ArrayDynamicType, [193](#)
- assert_liveliness
 - com::toc::coredx::DDS::DataWriter, [30](#)
 - com::toc::coredx::DDS::DomainParticipant, [39](#)
- attach_condition
 - com::toc::coredx::DDS::WaitSet, [74](#)
- BY_RECEPTION_TIMESTAMP_DESTINATIONORDER_QOS
 - com::toc::coredx::DDS::DestinationOrderQosPolicyKind, [139](#)
- begin_access
 - com::toc::coredx::DDS::Subscriber, [65](#)
- begin_coherent_changes
 - com::toc::coredx::DDS::Publisher, [52](#)
- bind
 - com::toc::coredx::DDS::rpc::ServiceProxy, [256](#)
- BooleanDynamicType, [194](#)
- broadcast_address
 - com::toc::coredx::DDS::UdpTransportConfig, [224](#)
- builtin_wait_for_acknowledgments
 - com::toc::coredx::DDS::DomainParticipant, [39](#)
- CLOSED
 - com::toc::coredx::DDS::rpc::ServiceStatus, [267](#)
- CharDynamicType, [195](#)
- cleanup
 - DDS Quality of Service, [6](#)
- ClientEndpoint
 - com::toc::coredx::DDS::rpc::ClientEndpoint, [229](#)
- ClientEndpoint< TReq extends RequestType, TRep extends ReplyType >, [229](#)
- ClientParams, [231](#)
 - com::toc::coredx::DDS::rpc::ClientParams, [231](#)
- close
 - com::toc::coredx::DDS::rpc::RPCEntity, [264](#)
 - com::toc::coredx::DDS::rpc::Replier, [238](#)
 - com::toc::coredx::DDS::rpc::ServiceEndpoint, [251](#)
 - com::toc::coredx::DDS::rpc::ServiceProxy, [256](#)
- com.toc.coredx.DDS.rpc, [287](#)
- com::toc::coredx::DDS::Condition
 - get_trigger_value, [21](#)
- com::toc::coredx::DDS::ContentFilteredTopic
 - get_expression_parameters, [22](#)
 - get_related_topic, [22](#)
 - set_expression_parameters, [22](#)
- com::toc::coredx::DDS::DataReader
 - create_querycondition, [25](#)
 - create_readcondition, [25](#)
 - delete_contained_entities, [25](#)
 - delete_readcondition, [25](#)
 - enable, [26](#)
 - get_listener, [26](#)
 - get_liveliness_changed_status, [26](#)
 - get_matched_publication_data, [26](#)
 - get_matched_publications, [26](#)
 - get_qos, [27](#)
 - get_requested_deadline_missed_status, [27](#)
 - get_requested_incompatible_qos_status, [27](#)
 - get_sample_lost_status, [27](#)
 - get_sample_rejected_status, [27](#)
 - get_subscriber, [27](#)
 - get_subscription_matched_status, [27](#)
 - get_topicdescription, [27](#)
 - set_listener, [27](#)
 - set_qos, [27](#)

- wait_for_historical_data, 28
- com::toc::coredx::DDS::DataReaderListener
 - get_nil_mask, 153
 - on_data_available, 153
 - on_liveliness_changed, 154
 - on_requested_deadline_missed, 154
 - on_requested_incompatible_qos, 154
 - on_sample_lost, 154
 - on_sample_rejected, 154
 - on_subscription_matched, 154
- com::toc::coredx::DDS::DataReaderQos
 - deadline, 90
 - destination_order, 90
 - durability, 90
 - entity_name, 90
 - history, 90
 - latency_budget, 90
 - liveliness, 90
 - logging, 90
 - ownership, 90
 - reader_data_lifecycle, 90
 - reliability, 91
 - representation, 91
 - resource_limits, 91
 - rpc, 91
 - rtps_reader, 91
 - time_based_filter, 91
 - type_consistency, 91
 - user_data, 91
- com::toc::coredx::DDS::DataRepresentationQosPolicy
 - value, 106
- com::toc::coredx::DDS::DataWriter
 - assert_liveliness, 30
 - enable, 30
 - get_listener, 30
 - get_liveliness_lost_status, 30
 - get_matched_subscription_data, 30
 - get_matched_subscriptions, 30
 - get_offered_deadline_missed_status, 31
 - get_offered_incompatible_qos_status, 31
 - get_publication_matched_status, 31
 - get_publisher, 31
 - get_qos, 31
 - get_topic, 31
 - set_listener, 31
 - set_qos, 31
 - wait_for_acknowledgments, 31
- com::toc::coredx::DDS::DataWriterListener
 - get_nil_mask, 155
 - on_liveliness_lost, 155
 - on_offered_deadline_missed, 155
 - on_offered_incompatible_qos, 155
 - on_publication_matched, 155
- com::toc::coredx::DDS::DataWriterQos
 - deadline, 92
 - destination_order, 92
 - durability, 93
 - durability_service, 93
 - entity_name, 93
 - history, 93
 - latency_budget, 93
 - lifespan, 93
 - liveliness, 93
 - logging, 93
 - ownership, 93
 - ownership_strength, 93
 - reliability, 93
 - representation, 94
 - resource_limits, 94
 - rpc, 94
 - rtps_writer, 94
 - transport_priority, 94
 - user_data, 94
 - writer_data_lifecycle, 94
- com::toc::coredx::DDS::DestinationOrderQosPolicyKind
 - BY_RECEPTION_TIMESTAMP_DESTINATIONORDER_QOS, 139
- com::toc::coredx::DDS::DiscoveryQosPolicyDiscoveryKind
 - PEER_DISCOVERY_QOS, 140
- com::toc::coredx::DDS::DomainParticipant
 - add_tcp_transport, 38
 - add_transport, 38
 - assert_liveliness, 39
 - builtin_wait_for_acknowledgments, 39
 - contains_entity, 39
 - create_contentfilteredtopic, 39
 - create_multitopic, 39
 - create_publisher, 39
 - create_subscriber, 40
 - create_topic, 40
 - delete_contained_entities, 40
 - delete_contentfilteredtopic, 40
 - delete_multitopic, 41
 - delete_publisher, 41
 - delete_subscriber, 41
 - delete_topic, 41
 - do_work, 41
 - enable, 41
 - find_topic, 42
 - get_builtin_subscriber, 42
 - get_current_time, 42
 - get_default_publisher_qos, 42
 - get_default_subscriber_qos, 43
 - get_default_topic_qos, 43
 - get_discovered_participant_data, 43
 - get_discovered_participants, 43
 - get_discovered_topic_data, 43
 - get_discovered_topics, 43

- get_domain_id, 43
- get_listener, 44
- get_qos, 44
- ignore_participant, 44
- ignore_publication, 44
- ignore_subscription, 44
- ignore_topic, 44
- lookup_topicdescription, 45
- register_type, 45
- set_default_publisher_qos, 45
- set_default_subscriber_qos, 45
- set_default_topic_qos, 45
- set_listener, 45
- set_qos, 46
- com::toc::coredx::DDS::DomainParticipantFactory
 - create_participant, 34
 - delete_participant, 35
 - get_default_participant_qos, 35
 - get_instance, 35
 - get_qos, 35
 - lookup_participant, 35
 - set_default_participant_qos, 35
 - set_qos, 36
- com::toc::coredx::DDS::DomainParticipantFactoryQos
 - entity_factory, 95
- com::toc::coredx::DDS::DomainParticipantListener
 - get_nil_mask, 157
 - on_data_available, 157
 - on_data_on_readers, 157
 - on_inconsistent_topic, 157
 - on_liveliness_changed, 158
 - on_liveliness_lost, 158
 - on_offered_deadline_missed, 158
 - on_offered_incompatible_qos, 158
 - on_publication_matched, 158
 - on_requested_deadline_missed, 158
 - on_requested_incompatible_qos, 159
 - on_sample_lost, 159
 - on_sample_rejected, 159
 - on_subscription_matched, 159
- com::toc::coredx::DDS::DomainParticipantQos
 - discovery, 96
 - entity_factory, 96
 - entity_name, 96
 - logging, 97
 - peer_participants, 97
 - properties, 97
 - thread_model, 97
 - user_data, 97
- com::toc::coredx::DDS::DurabilityQosPolicyKind
 - TRANSIENT_DURABILITY_QOS, 141
 - TRANSIENT_LOCAL_DURABILITY_QOS, 141
 - VOLATILE_DURABILITY_QOS, 141
- com::toc::coredx::DDS::DynamicType
 - release, 199
- com::toc::coredx::DDS::DynamicTypeTypeSupport
 - create_datareader, 200
 - create_datawriter, 200
 - getCTypeSupport, 200
 - register_type, 200
- com::toc::coredx::DDS::Entity
 - enable, 47
 - get_instance_handle, 47
 - get_status_changes, 47
 - get_statuscondition, 47
- com::toc::coredx::DDS::FooDataReader
 - get_key_value, 78
 - lookup_instance, 78
 - read, 78
 - read_instance, 79
 - read_next_instance, 79
 - read_next_instance_w_condition, 80
 - read_next_sample, 80
 - read_w_condition, 80
 - return_loan, 80
 - take, 81
 - take_instance, 81
 - take_next_instance, 82
 - take_next_instance_w_condition, 82
 - take_next_sample, 82
 - take_w_condition, 82
- com::toc::coredx::DDS::FooDataWriter
 - dispose, 84
 - dispose_w_timestamp, 84
 - get_key_value, 85
 - lookup_instance, 85
 - register_instance, 85
 - register_instance_w_timestamp, 85
 - unregister_instance, 85
 - unregister_instance_w_timestamp, 85
 - write, 85
 - write_w_timestamp, 86
- com::toc::coredx::DDS::FooTypeSupport
 - get_type_name, 87
 - register_type, 87
- com::toc::coredx::DDS::GuardCondition
 - GuardCondition, 48
- com::toc::coredx::DDS::HistoryQosPolicyKind
 - KEEP_LAST_HISTORY_QOS, 142
- com::toc::coredx::DDS::InconsistentTopicStatus
 - get_total_count, 165
 - get_total_count_change, 165
- com::toc::coredx::DDS::LivelinessChangedStatus
 - get_alive_count, 166
 - get_alive_count_change, 166
 - get_last_publication_handle, 166
 - get_not_alive_count, 166
 - get_not_alive_count_change, 166

- com::toc::coredx::DDS::LivelinessLostStatus
 - get_total_count, 167
 - get_total_count_change, 167
 - com::toc::coredx::DDS::LivelinessQosPolicyKind
 - AUTOMATIC_LIVELINESS_QOS, 143
 - MANUAL_BY_PARTICIPANT_LIVELINESS_QOS, 143
 - com::toc::coredx::DDS::LmtTransportConfig
 - debug_flags, 216
 - get_default_config, 216
 - get_env_config, 216
 - max_rx_buf_size, 216
 - max_tx_size, 216
 - so_rcvbuf, 217
 - so_sndbuf, 217
 - com::toc::coredx::DDS::LoanedSamples
 - length, 177
 - release, 177
 - return_loan, 177
 - size, 178
 - com::toc::coredx::DDS::Locator
 - addr, 49
 - com::toc::coredx::DDS::MultiTopic
 - get_name, 50
 - get_participant, 50
 - get_type_name, 50
 - com::toc::coredx::DDS::OfferedDeadlineMissedStatus
 - get_last_instance_handle, 168
 - get_total_count, 168
 - get_total_count_change, 168
 - com::toc::coredx::DDS::OfferedIncompatibleQosStatus
 - get_last_policy_id, 169
 - get_policies, 169
 - get_total_count, 169
 - get_total_count_change, 169
 - com::toc::coredx::DDS::OwnershipQosPolicyKind
 - SHARED_OWNERSHIP_QOS, 144
 - com::toc::coredx::DDS::ParticipantLocator
 - participant_id, 51
 - participant_id_max, 51
 - participant_locator, 51
 - com::toc::coredx::DDS::PeerParticipantQosPolicy
 - strict_match, 122
 - value, 122
 - com::toc::coredx::DDS::PresentationQosPolicyAccessScopeKind
 - INSTANCE_PRESENTATION_QOS, 145
 - TOPIC_PRESENTATION_QOS, 145
 - com::toc::coredx::DDS::Property_t
 - name, 182
 - propagate, 182
 - value, 182
 - com::toc::coredx::DDS::PropertyQosPolicy
 - value, 124
 - com::toc::coredx::DDS::PublicationMatchedStatus
 - get_current_count, 170
 - get_current_count_change, 170
 - get_last_subscription_handle, 170
 - get_total_count, 170
 - get_total_count_change, 170
 - com::toc::coredx::DDS::Publisher
 - begin_coherent_changes, 52
 - copy_from_topic_qos, 52
 - create_datawriter, 53
 - delete_contained_entities, 53
 - delete_datawriter, 53
 - enable, 53
 - end_coherent_changes, 53
 - get_default_datawriter_qos, 54
 - get_listener, 54
 - get_participant, 54
 - get_qos, 54
 - lookup_datawriter, 54
 - resume_publications, 54
 - set_default_datawriter_qos, 54
 - set_listener, 54
 - set_qos, 55
 - suspend_publications, 55
 - wait_for_acknowledgments, 55
 - com::toc::coredx::DDS::PublisherListener
 - get_nil_mask, 160
 - on_liveliness_lost, 160
 - on_offered_deadline_missed, 160
 - on_offered_incompatible_qos, 160
 - on_publication_matched, 160
 - com::toc::coredx::DDS::PublisherQos
 - entity_factory, 98
 - entity_name, 98
 - group_data, 98
 - logging, 98
 - partition, 98
 - presentation, 99
 - com::toc::coredx::DDS::QueryCondition
 - get_query_expression, 57
 - get_query_parameters, 57
 - set_query_parameters, 58
 - com::toc::coredx::DDS::ReadCondition
 - get_datareader, 59
 - get_instance_state_mask, 59
 - get_sample_state_mask, 59
 - get_view_state_mask, 59
 - com::toc::coredx::DDS::ReliabilityQosPolicy
 - kind, 126
 - max_blocking_time, 126
 - com::toc::coredx::DDS::RequestedDeadlineMissedStatus
 - get_last_instance_handle, 171
 - get_total_count, 171
 - get_total_count_change, 171
 - com::toc::coredx::DDS::RequestedIncompatibleQosStatus
-

- get_last_policy_id, 172
- get_policies, 172
- get_total_count, 172
- get_total_count_change, 172
- com::toc::coredx::DDS::ResourceLimitsQosPolicy
 - preallocate_instances, 127
 - preallocate_samples, 127
- com::toc::coredx::DDS::RpcQosPolicy
 - related_entity_guid, 128
 - service_instance_name, 128
 - topic_aliases, 128
- com::toc::coredx::DDS::SampleInfo
 - absolute_generation_rank, 61
 - disposed_generation_count, 61
 - generation_rank, 61
 - instance_handle, 61
 - instance_state, 62
 - no_writers_generation_count, 62
 - publication_handle, 62
 - sample_rank, 62
 - sample_state, 62
 - source_timestamp, 62
 - valid_data, 62
 - view_state, 62
- com::toc::coredx::DDS::SampleLostStatus
 - get_total_count, 173
 - get_total_count_change, 173
- com::toc::coredx::DDS::SampleRejectedStatus
 - get_last_instance_handle, 174
 - get_last_reason, 174
 - get_total_count, 174
 - get_total_count_change, 174
- com::toc::coredx::DDS::StatusCondition
 - get_enabled_statuses, 64
 - get_entity, 64
 - set_enabled_statuses, 64
- com::toc::coredx::DDS::Subscriber
 - begin_access, 65
 - copy_from_topic_qos, 65
 - create_datareader, 66
 - delete_contained_entities, 66
 - delete_datareader, 66
 - enable, 66
 - end_access, 67
 - get_datareaders, 67
 - get_default_datareader_qos, 67
 - get_listener, 67
 - get_participant, 67
 - get_qos, 67
 - lookup_datareader, 67
 - set_default_datareader_qos, 68
 - set_listener, 68
 - set_qos, 68
- com::toc::coredx::DDS::SubscriberListener
 - get_nil_mask, 162
 - on_data_available, 162
 - on_data_on_readers, 162
 - on_liveliness_changed, 162
 - on_requested_deadline_missed, 162
 - on_requested_incompatible_qos, 163
 - on_sample_lost, 163
 - on_sample_rejected, 163
 - on_subscription_matched, 163
- com::toc::coredx::DDS::SubscriberQos
 - entity_factory, 100
 - entity_name, 100
 - group_data, 100
 - logging, 100
 - partition, 100
 - presentation, 101
- com::toc::coredx::DDS::SubscriptionMatchedException
 - get_current_count, 175
 - get_current_count_change, 175
 - get_last_publication_handle, 175
 - get_total_count, 175
 - get_total_count_change, 175
- com::toc::coredx::DDS::TcpTransportConfig
 - dynamic_interfaces, 219
 - get_default_config, 219
 - get_env_config, 219
 - interfaces, 219
 - participant_index, 219
 - tx_max_packet_size, 220
- com::toc::coredx::DDS::Topic
 - enable, 69
 - get_inconsistent_topic_status, 70
 - get_listener, 70
 - get_qos, 70
 - set_listener, 70
 - set_qos, 70
- com::toc::coredx::DDS::TopicDescription
 - get_name, 71
 - get_participant, 71
 - get_type_name, 71
- com::toc::coredx::DDS::TopicListener
 - get_nil_mask, 164
 - on_inconsistent_topic, 164
- com::toc::coredx::DDS::TopicQos
 - deadline, 102
 - destination_order, 102
 - durability, 102
 - durability_service, 103
 - entity_name, 103
 - history, 103
 - latency_budget, 103
 - lifespan, 103
 - liveliness, 103
 - logging, 103

- ownership, [103](#)
- reliability, [103](#)
- resource_limits, [103](#)
- topic_data, [103](#)
- transport_priority, [104](#)
- com::toc::coredx::DDS::TypeCodeKind
 - coreToEnum, [147](#)
 - TYPECODE_ALIAS, [147](#)
 - TYPECODE_ARRAY, [147](#)
 - TYPECODE_BOOLEAN, [148](#)
 - TYPECODE_CHAR, [148](#)
 - TYPECODE_DOUBLE, [148](#)
 - TYPECODE_ENUM, [148](#)
 - TYPECODE_FLOAT, [148](#)
 - TYPECODE_LONGDOUBLE, [148](#)
 - TYPECODE_LONGLONG, [148](#)
 - TYPECODE_LONG, [148](#)
 - TYPECODE_OCTET, [148](#)
 - TYPECODE_SEQUENCE, [148](#)
 - TYPECODE_SHORT, [148](#)
 - TYPECODE_STRING, [149](#)
 - TYPECODE_STRUCT, [149](#)
 - TYPECODE_ULONGLONG, [149](#)
 - TYPECODE_ULONG, [149](#)
 - TYPECODE_UNION, [149](#)
 - TYPECODE_USHORT, [149](#)
 - TYPECODE_VALUE, [149](#)
 - TYPECODE_WCHAR, [149](#)
 - TYPECODE_WSTRING, [149](#)
- com::toc::coredx::DDS::TypeConsistencyEnforcementQosPolicy
 - kind, [136](#)
- com::toc::coredx::DDS::TypeConsistencyKind
 - ALLOW_TYPE_COERCION, [150](#)
 - DISALLOW_TYPE_COERCION, [150](#)
- com::toc::coredx::DDS::TypeSupport
 - create_datareader, [72](#)
 - create_datawriter, [72](#)
 - get_fully_qualified_typename, [72](#)
 - get_type, [72](#)
 - get_typename, [72](#)
 - getCTypeSupport, [73](#)
 - register_type, [73](#)
- com::toc::coredx::DDS::UdpTransportConfig
 - advertise_meta_multicast, [224](#)
 - advertise_user_multicast, [224](#)
 - broadcast_address, [224](#)
 - do_meta_broadcast, [224](#)
 - dynamic_interfaces, [224](#)
 - get_default_config, [223](#)
 - get_env_config, [223](#)
 - interfaces, [224](#)
 - meta_multicast_address_v4, [224](#)
 - meta_multicast_address_v6, [224](#)
 - multicast_ttl, [224](#)
 - participant_index, [224](#)
 - rx_init_buffer_size, [224](#)
 - rx_max_buffer_size, [225](#)
 - rx_meta_multicast, [225](#)
 - rx_user_multicast, [225](#)
 - so_rcvbuf, [225](#)
 - so_sndbuf, [225](#)
 - tx_max_packet_size, [225](#)
 - tx_meta_multicast, [225](#)
 - tx_meta_unicast, [225](#)
 - use_ipv4, [225](#)
 - use_ipv6, [225](#)
 - user_multicast_address_v4, [225](#)
 - user_multicast_address_v6, [226](#)
- com::toc::coredx::DDS::WaitSet
 - attach_condition, [74](#)
 - destroy, [74](#)
 - detach_condition, [74](#)
 - get_conditions, [74](#)
 - wait, [75](#)
 - WaitSet, [74](#)
- com::toc::coredx::DDS::rpc::ClientEndpoint
 - ClientEndpoint, [229](#)
 - get_client_params, [230](#)
 - receive_reply, [230](#)
 - send_request, [230](#)
- com::toc::coredx::DDS::rpc::ClientParams
 - ClientParams, [231](#)
 - datareader_qos, [231](#)
 - datawriter_qos, [232](#)
 - domain_participant, [232](#)
 - instance_name, [232](#)
 - publisher, [232](#)
 - reply_topic_name, [232](#)
 - request_topic_name, [232](#), [233](#)
 - service_name, [233](#)
 - subscriber, [233](#)
- com::toc::coredx::DDS::rpc::RPCEntity
 - close, [264](#)
 - is_null, [264](#)
- com::toc::coredx::DDS::rpc::Replier
 - close, [238](#)
 - get_replier_params, [238](#)
 - get_reply_datawriter, [238](#)
 - get_request_datareader, [238](#)
 - is_null, [238](#)
 - read_request, [238](#)
 - read_requests, [238](#)
 - receive_nondata_samples, [239](#)
 - receive_request, [239](#)
 - receive_requests, [239](#)
 - Replier, [237](#)
 - send_reply, [239](#)
 - take_request, [239](#)

- take_requests, 239
 - wait_for_requests, 239, 240
 - com::toc::coredx::DDS::rpc::ReplierListener
 - on_request_available, 262
 - com::toc::coredx::DDS::rpc::ReplierParams
 - datareader_qos, 235
 - datawriter_qos, 235
 - domain_participant, 235
 - instance_name, 235
 - publisher, 235
 - replier_listener, 235
 - ReplierParams, 234
 - reply_topic_name, 236
 - request_topic_name, 236
 - service_name, 236
 - simple_replier_listener, 236
 - subscriber, 236
 - com::toc::coredx::DDS::rpc::Requester
 - get_reply_datareader, 246
 - get_request_datawriter, 246
 - get_requester_params, 246
 - read_replies, 246
 - read_reply, 246, 247
 - receive_nondata_samples, 247
 - receive_replies, 247
 - receive_reply, 247, 248
 - Requester, 246
 - send_request, 248
 - send_request_oneway, 248
 - take_replies, 248
 - take_reply, 248, 249
 - wait_for_replies, 249
 - com::toc::coredx::DDS::rpc::RequesterListener
 - on_reply_available, 263
 - com::toc::coredx::DDS::rpc::RequesterParams
 - datareader_qos, 242
 - datawriter_qos, 242
 - domain_participant, 242
 - publisher, 242
 - reply_topic_name, 242
 - request_topic_name, 243
 - requester_listener, 243
 - RequesterParams, 241
 - service_name, 243
 - simple_requester_listener, 243
 - subscriber, 243, 244
 - com::toc::coredx::DDS::rpc::ServiceEndpoint
 - close, 251
 - get_service_params, 251
 - is_null, 251
 - pause, 252
 - resume, 252
 - ServiceEndpoint, 251
 - status, 252
 - com::toc::coredx::DDS::rpc::ServiceParams
 - datareader_qos, 253
 - datawriter_qos, 254
 - domain_participant, 254
 - instance_name, 254
 - publisher, 254
 - reply_topic_name, 254
 - request_topic_name, 254, 255
 - service_name, 255
 - ServiceParams, 253
 - subscriber, 255
 - com::toc::coredx::DDS::rpc::ServiceProxy
 - bind, 256
 - close, 256
 - get_bound_instance_name, 257
 - get_discovered_service_instances, 257
 - is_bound, 257
 - is_null, 257
 - unbind, 257
 - wait_for_service, 257, 258
 - wait_for_services, 258, 259
 - com::toc::coredx::DDS::rpc::ServiceStatus
 - CLOSED, 267
 - PAUSED, 267
 - RUNNING, 267
 - com::toc::coredx::DDS::rpc::SimpleReplierListener
 - process_request, 265
 - com::toc::coredx::DDS::rpc::SimpleRequesterListener
 - process_reply, 266
 - Condition, 21
 - Constants, 191
 - contains_entity
 - com::toc::coredx::DDS::DomainParticipant, 39
 - ContentFilteredTopic, 22
 - copy_from_topic_qos
 - com::toc::coredx::DDS::Publisher, 52
 - com::toc::coredx::DDS::Subscriber, 65
 - CoreDX DDS DynamicType, 18
 - CoreDX DDS RPC, 17
 - CoreDX DDS Transports, 16
 - coreToEnum
 - com::toc::coredx::DDS::TypeCodeKind, 147
 - create_contentfilteredtopic
 - com::toc::coredx::DDS::DomainParticipant, 39
 - create_datareader
 - com::toc::coredx::DDS::DynamicTypeTypeSupport, 200
 - com::toc::coredx::DDS::Subscriber, 66
 - com::toc::coredx::DDS::TypeSupport, 72
 - create_datawriter
 - com::toc::coredx::DDS::DynamicTypeTypeSupport, 200
 - com::toc::coredx::DDS::Publisher, 53
 - com::toc::coredx::DDS::TypeSupport, 72
-

- create_multitopic
 - com::toc::coredx::DDS::DomainParticipant, 39
- create_participant
 - com::toc::coredx::DDS::DomainParticipantFactory, 34
- create_publisher
 - com::toc::coredx::DDS::DomainParticipant, 39
- create_querycondition
 - com::toc::coredx::DDS::DataReader, 25
- create_readcondition
 - com::toc::coredx::DDS::DataReader, 25
- create_subscriber
 - com::toc::coredx::DDS::DomainParticipant, 40
- create_topic
 - com::toc::coredx::DDS::DomainParticipant, 40
- DATA_AVAILABLE_STATUS
 - DDS Status Structures, 15
- DATA_ON_READERS_STATUS
 - DDS Status Structures, 15
- DATA_REPRESENTATION_QOS_POLICY_ID
 - DDS Quality of Service, 8
- DATAREADER_QOS_DEFAULT
 - DDS Quality of Service, 8
- DATAWRITER_QOS_DEFAULT
 - DDS Quality of Service, 8
- DDS Builtin Entities and Data Types, 19
- DDS Conditions, 12
- DDS Conditions, Listeners, and WaitSets, 10
 - ALIVE_INSTANCE_STATE, 10
 - ANY_INSTANCE_STATE, 10
 - ANY_SAMPLE_STATE, 10
 - ANY_VIEW_STATE, 10
 - NEW_VIEW_STATE, 10
 - NOT_ALIVE_DISPOSED_INSTANCE_STATE, 10
 - NOT_ALIVE_INSTANCE_STATE, 10
 - NOT_ALIVE_NO_WRITERS_INSTANCE_STATE, 10
 - NOT_NEW_VIEW_STATE, 10
 - NOT_READ_SAMPLE_STATE, 10
 - READ_SAMPLE_STATE, 10
- DDS Entities, 4
- DDS Listeners, 11
- DDS Quality of Service, 5
 - cleanup, 6
 - DATA_REPRESENTATION_QOS_POLICY_ID, 8
 - DATAREADER_QOS_DEFAULT, 8
 - DATAWRITER_QOS_DEFAULT, 8
 - DEADLINE_QOS_POLICY_ID, 8
 - DESTINATIONORDER_QOS_POLICY_ID, 8
 - DURABILITY_QOS_POLICY_ID, 8
 - DURABILITYSERVICE_QOS_POLICY_ID, 8
 - ENTITYFACTORY_QOS_POLICY_ID, 8
 - GROUPDATA_QOS_POLICY_ID, 8
 - get_datareader_qos, 6
 - get_datawriter_qos, 6
 - get_participant_qos, 7
 - get_participantfactory_qos, 7
 - get_publisher_qos, 7
 - get_subscriber_qos, 7
 - get_topic_qos, 7
 - HISTORY_QOS_POLICY_ID, 8
 - LATENCYBUDGET_QOS_POLICY_ID, 8
 - LIFESPAN_QOS_POLICY_ID, 8
 - LIVELINESS_QOS_POLICY_ID, 8
 - OWNERSHIP_QOS_POLICY_ID, 8
 - OWNERSHIPSTRENGTH_QOS_POLICY_ID, 8
 - PARTICIPANT_QOS_DEFAULT, 8
 - PARTITION_QOS_POLICY_ID, 8
 - PRESENTATION_QOS_POLICY_ID, 8
 - PUBLISHER_QOS_DEFAULT, 8
 - qos_policy_str, 7
 - QosProvider, 7
 - READERDATA_LIFECYCLE_QOS_POLICY_ID, 8
 - RELIABILITY_QOS_POLICY_ID, 8
 - RESOURCELIMITS_QOS_POLICY_ID, 8
 - SUBSCRIBER_QOS_DEFAULT, 8
 - TIMEBASED_FILTER_QOS_POLICY_ID, 8
 - TOPIC_QOS_DEFAULT, 8
 - TOPICDATA_QOS_POLICY_ID, 8
 - TRANSPORTPRIORITY_QOS_POLICY_ID, 9
 - USERDATA_QOS_POLICY_ID, 9
 - WRITERDATA_LIFECYCLE_QOS_POLICY_ID, 9
- DDS Status Structures, 14
 - ALL_STATUS, 15
 - DATA_AVAILABLE_STATUS, 15
 - DATA_ON_READERS_STATUS, 15
 - INCONSISTENT_TOPIC_STATUS, 15
 - LIVELINESS_CHANGED_STATUS, 15
 - LIVELINESS_LOST_STATUS, 15
 - NO_STATUS, 15
 - OFFERED_DEADLINE_MISSED_STATUS, 15
 - OFFERED_INCOMPATIBLE_QOS_STATUS, 15
 - PUBLICATION_MATCHED_STATUS, 15
 - REQUESTED_DEADLINE_MISSED_STATUS, 15
 - REQUESTED_INCOMPATIBLE_QOS_STATUS, 15
 - SAMPLE_LOST_STATUS, 15
 - SAMPLE_REJECTED_STATUS, 15
 - STATUS_MASK_NONE, 15
 - SUBSCRIPTION_MATCHED_STATUS, 15
- DDS WaitSets, 13
- DEADLINE_QOS_POLICY_ID
 - DDS Quality of Service, 8
- DESTINATIONORDER_QOS_POLICY_ID
 - DDS Quality of Service, 8
- DISALLOW_TYPE_COERCION
 - com::toc::coredx::DDS::TypeConsistencyKind, 150
- DURABILITY_QOS_POLICY_ID
 - DDS Quality of Service, 8

- DURABILITYSERVICE_QOS_POLICY_ID
 - DDS Quality of Service, 8
- DataReader, 24
- DataReaderListener, 153
- DataReaderQos, 89
- DataRepresentationQosPolicy, 106
- DataWriter, 29
- DataSetWriterListener, 155
- DataSetWriterQos, 92
- datareader_qos
 - com::toc::coredx::DDS::rpc::ClientParams, 231
 - com::toc::coredx::DDS::rpc::ReplierParams, 235
 - com::toc::coredx::DDS::rpc::RequesterParams, 242
 - com::toc::coredx::DDS::rpc::ServiceParams, 253
- datawriter_qos
 - com::toc::coredx::DDS::rpc::ClientParams, 232
 - com::toc::coredx::DDS::rpc::ReplierParams, 235
 - com::toc::coredx::DDS::rpc::RequesterParams, 242
 - com::toc::coredx::DDS::rpc::ServiceParams, 254
- deadline
 - com::toc::coredx::DDS::DataReaderQos, 90
 - com::toc::coredx::DDS::DataSetWriterQos, 92
 - com::toc::coredx::DDS::TopicQos, 102
- DeadlineQosPolicy, 105
- debug_flags
 - com::toc::coredx::DDS::LmtTransportConfig, 216
- delete_contained_entities
 - com::toc::coredx::DDS::DataReader, 25
 - com::toc::coredx::DDS::DomainParticipant, 40
 - com::toc::coredx::DDS::Publisher, 53
 - com::toc::coredx::DDS::Subscriber, 66
- delete_contentfilteredtopic
 - com::toc::coredx::DDS::DomainParticipant, 40
- delete_datareader
 - com::toc::coredx::DDS::Subscriber, 66
- delete_datawriter
 - com::toc::coredx::DDS::Publisher, 53
- delete_multitopic
 - com::toc::coredx::DDS::DomainParticipant, 41
- delete_participant
 - com::toc::coredx::DDS::DomainParticipantFactory, 35
- delete_publisher
 - com::toc::coredx::DDS::DomainParticipant, 41
- delete_readcondition
 - com::toc::coredx::DDS::DataReader, 25
- delete_subscriber
 - com::toc::coredx::DDS::DomainParticipant, 41
- delete_topic
 - com::toc::coredx::DDS::DomainParticipant, 41
- destination_order
 - com::toc::coredx::DDS::DataReaderQos, 90
 - com::toc::coredx::DDS::DataSetWriterQos, 92
 - com::toc::coredx::DDS::TopicQos, 102
- DestinationOrderQosPolicy, 107
- DestinationOrderQosPolicyKind, 139
- destroy
 - com::toc::coredx::DDS::WaitSet, 74
- detach_condition
 - com::toc::coredx::DDS::WaitSet, 74
- discovery
 - com::toc::coredx::DDS::DomainParticipantQos, 96
- DiscoveryQosPolicy, 108
- DiscoveryQosPolicyDiscoveryKind, 140
- dispose
 - com::toc::coredx::DDS::FooDataSetWriter, 84
- dispose_w_timestamp
 - com::toc::coredx::DDS::FooDataSetWriter, 84
- disposed_generation_count
 - com::toc::coredx::DDS::SampleInfo, 61
- do_meta_broadcast
 - com::toc::coredx::DDS::UdpTransportConfig, 224
- do_work
 - com::toc::coredx::DDS::DomainParticipant, 41
- domain_participant
 - com::toc::coredx::DDS::rpc::ClientParams, 232
 - com::toc::coredx::DDS::rpc::ReplierParams, 235
 - com::toc::coredx::DDS::rpc::RequesterParams, 242
 - com::toc::coredx::DDS::rpc::ServiceParams, 254
- DomainEntity, 33
- DomainId_t, 179
- DomainParticipant, 37
- DomainParticipantFactory, 34
- DomainParticipantFactoryQos, 95
- DomainParticipantListener, 157
- DomainParticipantQos, 96
- DoubleDynamicType, 196
- durability
 - com::toc::coredx::DDS::DataReaderQos, 90
 - com::toc::coredx::DDS::DataSetWriterQos, 93
 - com::toc::coredx::DDS::TopicQos, 102
- durability_service
 - com::toc::coredx::DDS::DataSetWriterQos, 93
 - com::toc::coredx::DDS::TopicQos, 103
- DurabilityQosPolicy, 109
- DurabilityQosPolicyKind, 141
- DurabilityServiceQosPolicy, 110
- Duration_t, 180
- dynamic_interfaces
 - com::toc::coredx::DDS::TcpTransportConfig, 219
 - com::toc::coredx::DDS::UdpTransportConfig, 224
- DynamicType, 199
- DynamicTypeDataSetReader, 197
- DynamicTypeDataSetWriter, 198
- DynamicTypeTypeSupport, 200
- ENTITYFACTORY_QOS_POLICY_ID
 - DDS Quality of Service, 8

- enable
 - com::toc::coredx::DDS::DataReader, 26
 - com::toc::coredx::DDS::DataWriter, 30
 - com::toc::coredx::DDS::DomainParticipant, 41
 - com::toc::coredx::DDS::Entity, 47
 - com::toc::coredx::DDS::Publisher, 53
 - com::toc::coredx::DDS::Subscriber, 66
 - com::toc::coredx::DDS::Topic, 69
 - end_access
 - com::toc::coredx::DDS::Subscriber, 67
 - end_coherent_changes
 - com::toc::coredx::DDS::Publisher, 53
 - Entity, 47
 - entity_factory
 - com::toc::coredx::DDS::DomainParticipantFactoryQos, 95
 - com::toc::coredx::DDS::DomainParticipantQos, 96
 - com::toc::coredx::DDS::PublisherQos, 98
 - com::toc::coredx::DDS::SubscriberQos, 100
 - entity_name
 - com::toc::coredx::DDS::DataReaderQos, 90
 - com::toc::coredx::DDS::DataWriterQos, 93
 - com::toc::coredx::DDS::DomainParticipantQos, 96
 - com::toc::coredx::DDS::PublisherQos, 98
 - com::toc::coredx::DDS::SubscriberQos, 100
 - com::toc::coredx::DDS::TopicQos, 103
 - EntityFactoryQosPolicy, 111
 - EntityNameQosPolicy, 112
 - EnumDynamicType, 202
 - find_topic
 - com::toc::coredx::DDS::DomainParticipant, 42
 - FloatDynamicType, 203
 - FooDataReader, 77
 - FooDataWriter, 84
 - FooTypeSupport, 87
 - GROUPDATA_QOS_POLICY_ID
 - DDS Quality of Service, 8
 - generation_rank
 - com::toc::coredx::DDS::SampleInfo, 61
 - get_alive_count
 - com::toc::coredx::DDS::LivelinessChangedStatus, 166
 - get_alive_count_change
 - com::toc::coredx::DDS::LivelinessChangedStatus, 166
 - get_bound_instance_name
 - com::toc::coredx::DDS::rpc::ServiceProxy, 257
 - get_builtin_subscriber
 - com::toc::coredx::DDS::DomainParticipant, 42
 - get_client_params
 - com::toc::coredx::DDS::rpc::ClientEndpoint, 230
 - get_conditions
 - com::toc::coredx::DDS::WaitSet, 74
 - get_current_count
 - com::toc::coredx::DDS::PublicationMatchedStatus, 170
 - com::toc::coredx::DDS::SubscriptionMatchedStatus, 175
 - get_current_count_change
 - com::toc::coredx::DDS::PublicationMatchedStatus, 170
 - com::toc::coredx::DDS::SubscriptionMatchedStatus, 175
 - get_current_time
 - com::toc::coredx::DDS::DomainParticipant, 42
 - get_datareader
 - com::toc::coredx::DDS::ReadCondition, 59
 - get_datareader_qos
 - DDS Quality of Service, 6
 - get_datareaders
 - com::toc::coredx::DDS::Subscriber, 67
 - get_datawriter_qos
 - DDS Quality of Service, 6
 - get_default_config
 - com::toc::coredx::DDS::LmtTransportConfig, 216
 - com::toc::coredx::DDS::TcpTransportConfig, 219
 - com::toc::coredx::DDS::UdpTransportConfig, 223
 - get_default_datareader_qos
 - com::toc::coredx::DDS::Subscriber, 67
 - get_default_datawriter_qos
 - com::toc::coredx::DDS::Publisher, 54
 - get_default_participant_qos
 - com::toc::coredx::DDS::DomainParticipantFactory, 35
 - get_default_publisher_qos
 - com::toc::coredx::DDS::DomainParticipant, 42
 - get_default_subscriber_qos
 - com::toc::coredx::DDS::DomainParticipant, 43
 - get_default_topic_qos
 - com::toc::coredx::DDS::DomainParticipant, 43
 - get_discovered_participant_data
 - com::toc::coredx::DDS::DomainParticipant, 43
 - get_discovered_participants
 - com::toc::coredx::DDS::DomainParticipant, 43
 - get_discovered_service_instances
 - com::toc::coredx::DDS::rpc::ServiceProxy, 257
 - get_discovered_topic_data
 - com::toc::coredx::DDS::DomainParticipant, 43
 - get_discovered_topics
 - com::toc::coredx::DDS::DomainParticipant, 43
 - get_domain_id
 - com::toc::coredx::DDS::DomainParticipant, 43
 - get_enabled_statuses
 - com::toc::coredx::DDS::StatusCondition, 64
 - get_entity
 - com::toc::coredx::DDS::StatusCondition, 64
 - get_env_config
-

- com::toc::coredx::DDS::LmtTransportConfig, 216
- com::toc::coredx::DDS::TcpTransportConfig, 219
- com::toc::coredx::DDS::UdpTransportConfig, 223
- get_expression_parameters
 - com::toc::coredx::DDS::ContentFilteredTopic, 22
- get_fully_qualified_typename
 - com::toc::coredx::DDS::TypeSupport, 72
- get_inconsistent_topic_status
 - com::toc::coredx::DDS::Topic, 70
- get_instance
 - com::toc::coredx::DDS::DomainParticipantFactory, 35
- get_instance_handle
 - com::toc::coredx::DDS::Entity, 47
- get_instance_state_mask
 - com::toc::coredx::DDS::ReadCondition, 59
- get_key_value
 - com::toc::coredx::DDS::FooDataReader, 78
 - com::toc::coredx::DDS::FooDataWriter, 85
- get_last_instance_handle
 - com::toc::coredx::DDS::OfferedDeadlineMissedStatus, 168
 - com::toc::coredx::DDS::RequestedDeadlineMissedStatus, 171
 - com::toc::coredx::DDS::SampleRejectedStatus, 174
- get_last_policy_id
 - com::toc::coredx::DDS::OfferedIncompatibleQosStatus, 169
 - com::toc::coredx::DDS::RequestedIncompatibleQosStatus, 172
- get_last_publication_handle
 - com::toc::coredx::DDS::LivelinessChangedStatus, 166
 - com::toc::coredx::DDS::SubscriptionMatchedStatus, 175
- get_last_reason
 - com::toc::coredx::DDS::SampleRejectedStatus, 174
- get_last_subscription_handle
 - com::toc::coredx::DDS::PublicationMatchedStatus, 170
- get_listener
 - com::toc::coredx::DDS::DataReader, 26
 - com::toc::coredx::DDS::DataWriter, 30
 - com::toc::coredx::DDS::DomainParticipant, 44
 - com::toc::coredx::DDS::Publisher, 54
 - com::toc::coredx::DDS::Subscriber, 67
 - com::toc::coredx::DDS::Topic, 70
- get_liveliness_changed_status
 - com::toc::coredx::DDS::DataReader, 26
- get_liveliness_lost_status
 - com::toc::coredx::DDS::DataWriter, 30
- get_matched_publication_data
 - com::toc::coredx::DDS::DataReader, 26
- get_matched_publications
 - com::toc::coredx::DDS::DataReader, 26
- com::toc::coredx::DDS::DataReader, 26
- get_matched_subscription_data
 - com::toc::coredx::DDS::DataWriter, 30
- get_matched_subscriptions
 - com::toc::coredx::DDS::DataWriter, 30
- get_name
 - com::toc::coredx::DDS::MultiTopic, 50
 - com::toc::coredx::DDS::TopicDescription, 71
- get_nil_mask
 - com::toc::coredx::DDS::DataReaderListener, 153
 - com::toc::coredx::DDS::DataWriterListener, 155
 - com::toc::coredx::DDS::DomainParticipantListener, 157
 - com::toc::coredx::DDS::PublisherListener, 160
 - com::toc::coredx::DDS::SubscriberListener, 162
 - com::toc::coredx::DDS::TopicListener, 164
- get_not_alive_count
 - com::toc::coredx::DDS::LivelinessChangedStatus, 166
- get_not_alive_count_change
 - com::toc::coredx::DDS::LivelinessChangedStatus, 166
- get_offered_deadline_missed_status
 - com::toc::coredx::DDS::DataWriter, 31
- get_offered_incompatible_qos_status
 - com::toc::coredx::DDS::DataWriter, 31
- get_participant
 - com::toc::coredx::DDS::MultiTopic, 50
 - com::toc::coredx::DDS::Publisher, 54
 - com::toc::coredx::DDS::Subscriber, 67
 - com::toc::coredx::DDS::TopicDescription, 71
- get_participant_qos
 - DDS Quality of Service, 7
- get_participantfactory_qos
 - DDS Quality of Service, 7
- get_policies
 - com::toc::coredx::DDS::OfferedIncompatibleQosStatus, 169
 - com::toc::coredx::DDS::RequestedIncompatibleQosStatus, 172
- get_publication_matched_status
 - com::toc::coredx::DDS::DataWriter, 31
- get_publisher
 - com::toc::coredx::DDS::DataWriter, 31
- get_publisher_qos
 - DDS Quality of Service, 7
- get_qos
 - com::toc::coredx::DDS::DataReader, 27
 - com::toc::coredx::DDS::DataWriter, 31
 - com::toc::coredx::DDS::DomainParticipant, 44
 - com::toc::coredx::DDS::DomainParticipantFactory, 35
 - com::toc::coredx::DDS::Publisher, 54
 - com::toc::coredx::DDS::Subscriber, 67

- com::toc::coredx::DDS::Topic, 70
- get_query_expression
 - com::toc::coredx::DDS::QueryCondition, 57
- get_query_parameters
 - com::toc::coredx::DDS::QueryCondition, 57
- get_related_topic
 - com::toc::coredx::DDS::ContentFilteredTopic, 22
- get_replier_params
 - com::toc::coredx::DDS::rpc::Replier, 238
- get_reply_datareader
 - com::toc::coredx::DDS::rpc::Requester, 246
- get_reply_datawriter
 - com::toc::coredx::DDS::rpc::Replier, 238
- get_request_datareader
 - com::toc::coredx::DDS::rpc::Replier, 238
- get_request_datawriter
 - com::toc::coredx::DDS::rpc::Requester, 246
- get_requested_deadline_missed_status
 - com::toc::coredx::DDS::DataReader, 27
- get_requested_incompatible_qos_status
 - com::toc::coredx::DDS::DataReader, 27
- get_requester_params
 - com::toc::coredx::DDS::rpc::Requester, 246
- get_sample_lost_status
 - com::toc::coredx::DDS::DataReader, 27
- get_sample_rejected_status
 - com::toc::coredx::DDS::DataReader, 27
- get_sample_state_mask
 - com::toc::coredx::DDS::ReadCondition, 59
- get_service_params
 - com::toc::coredx::DDS::rpc::ServiceEndpoint, 251
- get_status_changes
 - com::toc::coredx::DDS::Entity, 47
- get_statuscondition
 - com::toc::coredx::DDS::Entity, 47
- get_subscriber
 - com::toc::coredx::DDS::DataReader, 27
- get_subscriber_qos
 - DDS Quality of Service, 7
- get_subscription_matched_status
 - com::toc::coredx::DDS::DataReader, 27
- get_topic
 - com::toc::coredx::DDS::DataWriter, 31
- get_topic_qos
 - DDS Quality of Service, 7
- get_topicdescription
 - com::toc::coredx::DDS::DataReader, 27
- get_total_count
 - com::toc::coredx::DDS::InconsistentTopicStatus, 165
 - com::toc::coredx::DDS::LivelinessLostStatus, 167
 - com::toc::coredx::DDS::OfferedDeadlineMissedStatus, 168
 - com::toc::coredx::DDS::OfferedIncompatibleQosStatus, history 169
 - com::toc::coredx::DDS::PublicationMatchedStatus, 170
 - com::toc::coredx::DDS::RequestedDeadlineMissedStatus, 171
 - com::toc::coredx::DDS::RequestedIncompatibleQosStatus, 172
 - com::toc::coredx::DDS::SampleLostStatus, 173
 - com::toc::coredx::DDS::SampleRejectedStatus, 174
 - com::toc::coredx::DDS::SubscriptionMatchedStatus, 175
- get_total_count_change
 - com::toc::coredx::DDS::InconsistentTopicStatus, 165
 - com::toc::coredx::DDS::LivelinessLostStatus, 167
 - com::toc::coredx::DDS::OfferedDeadlineMissedStatus, 168
 - com::toc::coredx::DDS::OfferedIncompatibleQosStatus, 169
 - com::toc::coredx::DDS::PublicationMatchedStatus, 170
 - com::toc::coredx::DDS::RequestedDeadlineMissedStatus, 171
 - com::toc::coredx::DDS::RequestedIncompatibleQosStatus, 172
 - com::toc::coredx::DDS::SampleLostStatus, 173
 - com::toc::coredx::DDS::SampleRejectedStatus, 174
 - com::toc::coredx::DDS::SubscriptionMatchedStatus, 175
- get_trigger_value
 - com::toc::coredx::DDS::Condition, 21
- get_type
 - com::toc::coredx::DDS::TypeSupport, 72
- get_type_name
 - com::toc::coredx::DDS::FooTypeSupport, 87
 - com::toc::coredx::DDS::MultiTopic, 50
 - com::toc::coredx::DDS::TopicDescription, 71
- get_typename
 - com::toc::coredx::DDS::TypeSupport, 72
- get_view_state_mask
 - com::toc::coredx::DDS::ReadCondition, 59
- getCTypeSupport
 - com::toc::coredx::DDS::DynamicTypeTypeSupport, 200
 - com::toc::coredx::DDS::TypeSupport, 73
- group_data
 - com::toc::coredx::DDS::PublisherQos, 98
 - com::toc::coredx::DDS::SubscriberQos, 100
- GroupDataQosPolicy, 113
- GuardCondition, 48
 - com::toc::coredx::DDS::GuardCondition, 48
- HISTORY_QOS_POLICY_ID
 - DDS Quality of Service, 8
- com::toc::coredx::DDS::DataReaderQos, 90

- com::toc::coredx::DDS::DataWriterQos, 93
 - com::toc::coredx::DDS::TopicQos, 103
- HistoryQosPolicy, 114
- HistoryQosPolicyKind, 142
- INCONSISTENT_TOPIC_STATUS
 - DDS Status Structures, 15
- INSTANCE_PRESENTATION_QOS
 - com::toc::coredx::DDS::PresentationQosPolicyAccessScopeKind, 145
- ignore_participant
 - com::toc::coredx::DDS::DomainParticipant, 44
- ignore_publication
 - com::toc::coredx::DDS::DomainParticipant, 44
- ignore_subscription
 - com::toc::coredx::DDS::DomainParticipant, 44
- ignore_topic
 - com::toc::coredx::DDS::DomainParticipant, 44
- InconsistentTopicStatus, 165
- instance_handle
 - com::toc::coredx::DDS::SampleInfo, 61
- instance_name
 - com::toc::coredx::DDS::rpc::ClientParams, 232
 - com::toc::coredx::DDS::rpc::ReplierParams, 235
 - com::toc::coredx::DDS::rpc::ServiceParams, 254
- instance_state
 - com::toc::coredx::DDS::SampleInfo, 62
- InstanceHandle_t, 181
- interfaces
 - com::toc::coredx::DDS::TcpTransportConfig, 219
 - com::toc::coredx::DDS::UdpTransportConfig, 224
- IpTransportInterface, 215
- is_bound
 - com::toc::coredx::DDS::rpc::ServiceProxy, 257
- is_null
 - com::toc::coredx::DDS::rpc::RPCEntity, 264
 - com::toc::coredx::DDS::rpc::Replier, 238
 - com::toc::coredx::DDS::rpc::ServiceEndpoint, 251
 - com::toc::coredx::DDS::rpc::ServiceProxy, 257
- KEEP_LAST_HISTORY_QOS
 - com::toc::coredx::DDS::HistoryQosPolicyKind, 142
- kind
 - com::toc::coredx::DDS::ReliabilityQosPolicy, 126
 - com::toc::coredx::DDS::TypeConsistencyEnforcementQosPolicy, 136
- LATENCYBUDGET_QOS_POLICY_ID
 - DDS Quality of Service, 8
- LIFESPAN_QOS_POLICY_ID
 - DDS Quality of Service, 8
- LIVELINESS_CHANGED_STATUS
 - DDS Status Structures, 15
- LIVELINESS_LOST_STATUS
 - DDS Status Structures, 15
- LIVELINESS_QOS_POLICY_ID
 - DDS Quality of Service, 8
- latency_budget
 - com::toc::coredx::DDS::DataReaderQos, 90
 - com::toc::coredx::DDS::DataWriterQos, 93
 - com::toc::coredx::DDS::TopicQos, 103
- LatencyBudgetQosPolicy, 115
- length
 - com::toc::coredx::DDS::LoanedSamples, 177
- lifespan
 - com::toc::coredx::DDS::DataWriterQos, 93
 - com::toc::coredx::DDS::TopicQos, 103
- LifespanQosPolicy, 116
- ListenerBase, 260
- liveliness
 - com::toc::coredx::DDS::DataReaderQos, 90
 - com::toc::coredx::DDS::DataWriterQos, 93
 - com::toc::coredx::DDS::TopicQos, 103
- LivelinessChangedStatus, 166
- LivelinessLostStatus, 167
- LivelinessQosPolicy, 117
- LivelinessQosPolicyKind, 143
- LmtTransport, 218
- LmtTransportConfig, 216
- LoanedSamples< T >, 177
- Locator, 49
- logging
 - com::toc::coredx::DDS::DataReaderQos, 90
 - com::toc::coredx::DDS::DataWriterQos, 93
 - com::toc::coredx::DDS::DomainParticipantQos, 97
 - com::toc::coredx::DDS::PublisherQos, 98
 - com::toc::coredx::DDS::SubscriberQos, 100
 - com::toc::coredx::DDS::TopicQos, 103
- LoggingQosPolicy, 118
- LongDynamicType, 204
- LongLongDynamicType, 205
- lookup_datareader
 - com::toc::coredx::DDS::Subscriber, 67
- lookup_datawriter
 - com::toc::coredx::DDS::Publisher, 54
- lookup_instance
 - com::toc::coredx::DDS::FooDataReader, 78
 - com::toc::coredx::DDS::FooDataWriter, 85
- lookup_participant
 - com::toc::coredx::DDS::DomainParticipantFactory, 35
- lookup_topicdescription
 - com::toc::coredx::DDS::DomainParticipant, 45
- MANUAL_BY_PARTICIPANT_LIVELINESS_QOS
 - com::toc::coredx::DDS::LivelinessQosPolicyKind, 143
- max_blocking_time
 - com::toc::coredx::DDS::ReliabilityQosPolicy, 126

- max_rx_buf_size
 - com::toc::coredx::DDS::LmtTransportConfig, 216
- max_tx_size
 - com::toc::coredx::DDS::LmtTransportConfig, 216
- meta_multicast_address_v4
 - com::toc::coredx::DDS::UdpTransportConfig, 224
- meta_multicast_address_v6
 - com::toc::coredx::DDS::UdpTransportConfig, 224
- MultiTopic, 50
- multicast_ttl
 - com::toc::coredx::DDS::UdpTransportConfig, 224
- NEW_VIEW_STATE
 - DDS Conditions, Listeners, and WaitSets, 10
- NO_STATUS
 - DDS Status Structures, 15
- NOT_ALIVE_DISPOSED_INSTANCE_STATE
 - DDS Conditions, Listeners, and WaitSets, 10
- NOT_ALIVE_INSTANCE_STATE
 - DDS Conditions, Listeners, and WaitSets, 10
- NOT_ALIVE_NO_WRITERS_INSTANCE_STATE
 - DDS Conditions, Listeners, and WaitSets, 10
- NOT_NEW_VIEW_STATE
 - DDS Conditions, Listeners, and WaitSets, 10
- NOT_READ_SAMPLE_STATE
 - DDS Conditions, Listeners, and WaitSets, 10
- name
 - com::toc::coredx::DDS::Property_t, 182
- no_writers_generation_count
 - com::toc::coredx::DDS::SampleInfo, 62
- OFFERED_DEADLINE_MISSED_STATUS
 - DDS Status Structures, 15
- OFFERED_INCOMPATIBLE_QOS_STATUS
 - DDS Status Structures, 15
- OWNERSHIP_QOS_POLICY_ID
 - DDS Quality of Service, 8
- OWNERSHIPSTRENGTH_QOS_POLICY_ID
 - DDS Quality of Service, 8
- OctetDynamicType, 206
- OfferedDeadlineMissedStatus, 168
- OfferedIncompatibleQosStatus, 169
- on_data_available
 - com::toc::coredx::DDS::DataReaderListener, 153
 - com::toc::coredx::DDS::DomainParticipantListener, 157
 - com::toc::coredx::DDS::SubscriberListener, 162
- on_data_on_readers
 - com::toc::coredx::DDS::DomainParticipantListener, 157
 - com::toc::coredx::DDS::SubscriberListener, 162
- on_inconsistent_topic
 - com::toc::coredx::DDS::DomainParticipantListener, 157
 - com::toc::coredx::DDS::TopicListener, 164
- on_liveliness_changed
 - com::toc::coredx::DDS::DataReaderListener, 154
 - com::toc::coredx::DDS::DomainParticipantListener, 158
 - com::toc::coredx::DDS::SubscriberListener, 162
- on_liveliness_lost
 - com::toc::coredx::DDS::DataWriterListener, 155
 - com::toc::coredx::DDS::DomainParticipantListener, 158
 - com::toc::coredx::DDS::PublisherListener, 160
- on_offered_deadline_missed
 - com::toc::coredx::DDS::DataWriterListener, 155
 - com::toc::coredx::DDS::DomainParticipantListener, 158
 - com::toc::coredx::DDS::PublisherListener, 160
- on_offered_incompatible_qos
 - com::toc::coredx::DDS::DataWriterListener, 155
 - com::toc::coredx::DDS::DomainParticipantListener, 158
 - com::toc::coredx::DDS::PublisherListener, 160
- on_publication_matched
 - com::toc::coredx::DDS::DataWriterListener, 155
 - com::toc::coredx::DDS::DomainParticipantListener, 158
 - com::toc::coredx::DDS::PublisherListener, 160
- on_reply_available
 - com::toc::coredx::DDS::rpc::RequesterListener, 263
- on_request_available
 - com::toc::coredx::DDS::rpc::ReplierListener, 262
- on_requested_deadline_missed
 - com::toc::coredx::DDS::DataReaderListener, 154
 - com::toc::coredx::DDS::DomainParticipantListener, 158
 - com::toc::coredx::DDS::SubscriberListener, 162
- on_requested_incompatible_qos
 - com::toc::coredx::DDS::DataReaderListener, 154
 - com::toc::coredx::DDS::DomainParticipantListener, 159
 - com::toc::coredx::DDS::SubscriberListener, 163
- on_sample_lost
 - com::toc::coredx::DDS::DataReaderListener, 154
 - com::toc::coredx::DDS::DomainParticipantListener, 159
 - com::toc::coredx::DDS::SubscriberListener, 163
- on_sample_rejected
 - com::toc::coredx::DDS::DataReaderListener, 154
 - com::toc::coredx::DDS::DomainParticipantListener, 159
 - com::toc::coredx::DDS::SubscriberListener, 163
- on_subscription_matched
 - com::toc::coredx::DDS::DataReaderListener, 154
 - com::toc::coredx::DDS::DomainParticipantListener, 159
 - com::toc::coredx::DDS::SubscriberListener, 163

- ownership
 - com::toc::coredx::DDS::DataReaderQos, 90
 - com::toc::coredx::DDS::DataWriterQos, 93
 - com::toc::coredx::DDS::TopicQos, 103
 - ownership_strength
 - com::toc::coredx::DDS::DataWriterQos, 93
 - OwnershipQosPolicy, 119
 - OwnershipQosPolicyKind, 144
 - OwnershipStrengthQosPolicy, 120

 - PARTICIPANT_QOS_DEFAULT
 - DDS Quality of Service, 8
 - PARTITION_QOS_POLICY_ID
 - DDS Quality of Service, 8
 - PAUSED
 - com::toc::coredx::DDS::rpc::ServiceStatus, 267
 - PEER_DISCOVERY_QOS
 - com::toc::coredx::DDS::DiscoveryQosPolicyDiscoveryKind, 140
 - PRESENTATION_QOS_POLICY_ID
 - DDS Quality of Service, 8
 - PUBLICATION_MATCHED_STATUS
 - DDS Status Structures, 15
 - PUBLISHER_QOS_DEFAULT
 - DDS Quality of Service, 8
 - participant_id
 - com::toc::coredx::DDS::ParticipantLocator, 51
 - participant_id_max
 - com::toc::coredx::DDS::ParticipantLocator, 51
 - participant_index
 - com::toc::coredx::DDS::TcpTransportConfig, 219
 - com::toc::coredx::DDS::UdpTransportConfig, 224
 - participant_locator
 - com::toc::coredx::DDS::ParticipantLocator, 51
 - ParticipantBuiltinTopicData, 186
 - ParticipantBuiltinTopicDataDataReader, 185
 - ParticipantLocator, 51
 - partition
 - com::toc::coredx::DDS::PublisherQos, 98
 - com::toc::coredx::DDS::SubscriberQos, 100
 - PartitionQosPolicy, 121
 - pause
 - com::toc::coredx::DDS::rpc::ServiceEndpoint, 252
 - peer_participants
 - com::toc::coredx::DDS::DomainParticipantQos, 97
 - PeerParticipantQosPolicy, 122
 - preallocate_instances
 - com::toc::coredx::DDS::ResourceLimitsQosPolicy, 127
 - preallocate_samples
 - com::toc::coredx::DDS::ResourceLimitsQosPolicy, 127
 - presentation
 - com::toc::coredx::DDS::PublisherQos, 99
 - com::toc::coredx::DDS::SubscriberQos, 101
 - PresentationQosPolicy, 123
 - PresentationQosPolicyAccessScopeKind, 145
 - process_reply
 - com::toc::coredx::DDS::rpc::SimpleRequesterListener, 266
 - process_request
 - com::toc::coredx::DDS::rpc::SimpleReplierListener, 265
 - propagate
 - com::toc::coredx::DDS::Property_t, 182
 - properties
 - com::toc::coredx::DDS::DomainParticipantQos, 97
 - Property_t, 182
 - PropertyQosPolicy, 124
 - publication_handle
 - com::toc::coredx::DDS::SampleInfo, 62
 - PublicationBuiltinTopicData, 188
 - PublicationBuiltinTopicDataDataReader, 187
 - PublicationMatchedStatus, 170
 - Publisher, 52
 - publisher
 - com::toc::coredx::DDS::rpc::ClientParams, 232
 - com::toc::coredx::DDS::rpc::ReplierParams, 235
 - com::toc::coredx::DDS::rpc::RequesterParams, 242
 - com::toc::coredx::DDS::rpc::ServiceParams, 254
 - PublisherListener, 160
 - PublisherQos, 98

 - qos_policy_str
 - DDS Quality of Service, 7
 - QosPolicyCount, 56
 - QosProvider, 151
 - DDS Quality of Service, 7
 - QueryCondition, 57

 - READ_SAMPLE_STATE
 - DDS Conditions, Listeners, and WaitSets, 10
 - READER_DATA_LIFECYCLE_QOS_POLICY_ID
 - DDS Quality of Service, 8
 - RELIABILITY_QOS_POLICY_ID
 - DDS Quality of Service, 8
 - REQUESTED_DEADLINE_MISSED_STATUS
 - DDS Status Structures, 15
 - REQUESTED_INCOMPATIBLE_QOS_STATUS
 - DDS Status Structures, 15
 - RESOURCELIMITS_QOS_POLICY_ID
 - DDS Quality of Service, 8
 - RPCEntity, 264
 - RTPSReaderQosPolicy, 129
 - RTPSWriterQosPolicy, 130
 - RUNNING
 - com::toc::coredx::DDS::rpc::ServiceStatus, 267
 - read
 - com::toc::coredx::DDS::FooDataReader, 78
-

- read_instance
 - com::toc::coredx::DDS::FooDataReader, [79](#)
 - read_next_instance
 - com::toc::coredx::DDS::FooDataReader, [79](#)
 - read_next_instance_w_condition
 - com::toc::coredx::DDS::FooDataReader, [80](#)
 - read_next_sample
 - com::toc::coredx::DDS::FooDataReader, [80](#)
 - read_replies
 - com::toc::coredx::DDS::rpc::Requester, [246](#)
 - read_reply
 - com::toc::coredx::DDS::rpc::Requester, [246](#), [247](#)
 - read_request
 - com::toc::coredx::DDS::rpc::Replier, [238](#)
 - read_requests
 - com::toc::coredx::DDS::rpc::Replier, [238](#)
 - read_w_condition
 - com::toc::coredx::DDS::FooDataReader, [80](#)
 - ReadCondition, [59](#)
 - reader_data_lifecycle
 - com::toc::coredx::DDS::DataReaderQos, [90](#)
 - ReaderDataLifecycleQosPolicy, [125](#)
 - receive_nondata_samples
 - com::toc::coredx::DDS::rpc::Replier, [239](#)
 - com::toc::coredx::DDS::rpc::Requester, [247](#)
 - receive_replies
 - com::toc::coredx::DDS::rpc::Requester, [247](#)
 - receive_reply
 - com::toc::coredx::DDS::rpc::ClientEndpoint, [230](#)
 - com::toc::coredx::DDS::rpc::Requester, [247](#), [248](#)
 - receive_request
 - com::toc::coredx::DDS::rpc::Replier, [239](#)
 - receive_requests
 - com::toc::coredx::DDS::rpc::Replier, [239](#)
 - register_instance
 - com::toc::coredx::DDS::FooDataWriter, [85](#)
 - register_instance_w_timestamp
 - com::toc::coredx::DDS::FooDataWriter, [85](#)
 - register_type
 - com::toc::coredx::DDS::DomainParticipant, [45](#)
 - com::toc::coredx::DDS::DynamicTypeTypeSupport, [200](#)
 - com::toc::coredx::DDS::FooTypeSupport, [87](#)
 - com::toc::coredx::DDS::TypeSupport, [73](#)
 - related_entity_guid
 - com::toc::coredx::DDS::RpcQosPolicy, [128](#)
 - release
 - com::toc::coredx::DDS::DynamicType, [199](#)
 - com::toc::coredx::DDS::LoanedSamples, [177](#)
 - reliability
 - com::toc::coredx::DDS::DataReaderQos, [91](#)
 - com::toc::coredx::DDS::DataWriterQos, [93](#)
 - com::toc::coredx::DDS::TopicQos, [103](#)
 - ReliabilityQosPolicy, [126](#)
 - ReliabilityQosPolicyKind, [146](#)
 - Replier
 - com::toc::coredx::DDS::rpc::Replier, [237](#)
 - Replier< TReq extends RequestType, TRep extends ReplyType >, [237](#)
 - replier_listener
 - com::toc::coredx::DDS::rpc::ReplierParams, [235](#)
 - ReplierBase, [261](#)
 - ReplierListener< TReq extends RequestType, TRep extends ReplyType >, [262](#)
 - ReplierParams, [234](#)
 - com::toc::coredx::DDS::rpc::ReplierParams, [234](#)
 - reply_topic_name
 - com::toc::coredx::DDS::rpc::ClientParams, [232](#)
 - com::toc::coredx::DDS::rpc::ReplierParams, [236](#)
 - com::toc::coredx::DDS::rpc::RequesterParams, [242](#)
 - com::toc::coredx::DDS::rpc::ServiceParams, [254](#)
 - representation
 - com::toc::coredx::DDS::DataReaderQos, [91](#)
 - com::toc::coredx::DDS::DataWriterQos, [94](#)
 - request_topic_name
 - com::toc::coredx::DDS::rpc::ClientParams, [232](#), [233](#)
 - com::toc::coredx::DDS::rpc::ReplierParams, [236](#)
 - com::toc::coredx::DDS::rpc::RequesterParams, [243](#)
 - com::toc::coredx::DDS::rpc::ServiceParams, [254](#), [255](#)
 - RequestedDeadlineMissedStatus, [171](#)
 - RequestedIncompatibleQosStatus, [172](#)
 - Requester
 - com::toc::coredx::DDS::rpc::Requester, [246](#)
 - Requester< TReq extends RequestType, TRep extends ReplyType >, [245](#)
 - requester_listener
 - com::toc::coredx::DDS::rpc::RequesterParams, [243](#)
 - RequesterListener< TReq extends RequestType, TRep extends ReplyType >, [263](#)
 - RequesterParams, [241](#)
 - com::toc::coredx::DDS::rpc::RequesterParams, [241](#)
 - resource_limits
 - com::toc::coredx::DDS::DataReaderQos, [91](#)
 - com::toc::coredx::DDS::DataWriterQos, [94](#)
 - com::toc::coredx::DDS::TopicQos, [103](#)
 - ResourceLimitsQosPolicy, [127](#)
 - resume
 - com::toc::coredx::DDS::rpc::ServiceEndpoint, [252](#)
 - resume_publications
 - com::toc::coredx::DDS::Publisher, [54](#)
 - return_loan
 - com::toc::coredx::DDS::FooDataReader, [80](#)
 - com::toc::coredx::DDS::LoanedSamples, [177](#)
 - rpc
 - com::toc::coredx::DDS::DataReaderQos, [91](#)
 - com::toc::coredx::DDS::DataWriterQos, [94](#)
 - RpcQosPolicy, [128](#)
-

- rtps_reader
 - com::toc::coredx::DDS::DataReaderQos, [91](#)
- rtps_writer
 - com::toc::coredx::DDS::DataWriterQos, [94](#)
- rx_init_buffer_size
 - com::toc::coredx::DDS::UdpTransportConfig, [224](#)
- rx_max_buffer_size
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
- rx_meta_multicast
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
- rx_user_multicast
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
- SAMPLE_LOST_STATUS
 - DDS Status Structures, [15](#)
- SAMPLE_REJECTED_STATUS
 - DDS Status Structures, [15](#)
- SHARED_OWNERSHIP_QOS
 - com::toc::coredx::DDS::OwnershipQosPolicyKind, [144](#)
- STATUS_MASK_NONE
 - DDS Status Structures, [15](#)
- SUBSCRIBER_QOS_DEFAULT
 - DDS Quality of Service, [8](#)
- SUBSCRIPTION_MATCHED_STATUS
 - DDS Status Structures, [15](#)
- sample_rank
 - com::toc::coredx::DDS::SampleInfo, [62](#)
- sample_state
 - com::toc::coredx::DDS::SampleInfo, [62](#)
- SampleInfo, [61](#)
- SampleLostStatus, [173](#)
- SampleRejectedStatus, [174](#)
- send_reply
 - com::toc::coredx::DDS::rpc::Replier, [239](#)
- send_request
 - com::toc::coredx::DDS::rpc::ClientEndpoint, [230](#)
 - com::toc::coredx::DDS::rpc::Requester, [248](#)
- send_request_oneway
 - com::toc::coredx::DDS::rpc::Requester, [248](#)
- SequenceDynamicType, [207](#)
- service_instance_name
 - com::toc::coredx::DDS::RpcQosPolicy, [128](#)
- service_name
 - com::toc::coredx::DDS::rpc::ClientParams, [233](#)
 - com::toc::coredx::DDS::rpc::ReplierParams, [236](#)
 - com::toc::coredx::DDS::rpc::RequesterParams, [243](#)
 - com::toc::coredx::DDS::rpc::ServiceParams, [255](#)
- ServiceEndpoint, [251](#)
 - com::toc::coredx::DDS::rpc::ServiceEndpoint, [251](#)
- ServiceParams, [253](#)
 - com::toc::coredx::DDS::rpc::ServiceParams, [253](#)
- ServiceProxy< TReq extends RequestType, TRep extends ReplyType >, [256](#)
- ServiceStatus, [267](#)
- set_default_datareader_qos
 - com::toc::coredx::DDS::Subscriber, [68](#)
- set_default_datawriter_qos
 - com::toc::coredx::DDS::Publisher, [54](#)
- set_default_participant_qos
 - com::toc::coredx::DDS::DomainParticipantFactory, [35](#)
- set_default_publisher_qos
 - com::toc::coredx::DDS::DomainParticipant, [45](#)
- set_default_subscriber_qos
 - com::toc::coredx::DDS::DomainParticipant, [45](#)
- set_default_topic_qos
 - com::toc::coredx::DDS::DomainParticipant, [45](#)
- set_enabled_statuses
 - com::toc::coredx::DDS::StatusCondition, [64](#)
- set_expression_parameters
 - com::toc::coredx::DDS::ContentFilteredTopic, [22](#)
- set_listener
 - com::toc::coredx::DDS::DataReader, [27](#)
 - com::toc::coredx::DDS::DataWriter, [31](#)
 - com::toc::coredx::DDS::DomainParticipant, [45](#)
 - com::toc::coredx::DDS::Publisher, [54](#)
 - com::toc::coredx::DDS::Subscriber, [68](#)
 - com::toc::coredx::DDS::Topic, [70](#)
- set_qos
 - com::toc::coredx::DDS::DataReader, [27](#)
 - com::toc::coredx::DDS::DataWriter, [31](#)
 - com::toc::coredx::DDS::DomainParticipant, [46](#)
 - com::toc::coredx::DDS::DomainParticipantFactory, [36](#)
 - com::toc::coredx::DDS::Publisher, [55](#)
 - com::toc::coredx::DDS::Subscriber, [68](#)
 - com::toc::coredx::DDS::Topic, [70](#)
- set_query_parameters
 - com::toc::coredx::DDS::QueryCondition, [58](#)
- ShortDynamicType, [208](#)
- simple_replier_listener
 - com::toc::coredx::DDS::rpc::ReplierParams, [236](#)
- simple_requester_listener
 - com::toc::coredx::DDS::rpc::RequesterParams, [243](#)
- SimpleReplierListener< TReq extends RequestType, TRep extends ReplyType >, [265](#)
- SimpleRequesterListener< TRep extends ReplyType >, [266](#)
- size
 - com::toc::coredx::DDS::LoanedSamples, [178](#)
- so_rcvbuf
 - com::toc::coredx::DDS::LmtTransportConfig, [217](#)
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
- so_sndbuf
 - com::toc::coredx::DDS::LmtTransportConfig, [217](#)
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
- source_timestamp

- com::toc::coredx::DDS::SampleInfo, 62
- status
 - com::toc::coredx::DDS::rpc::ServiceEndpoint, 252
- StatusCondition, 64
- strict_match
 - com::toc::coredx::DDS::PeerParticipantQosPolicy, 122
- StringDynamicType, 209
- StructDynamicType, 210
- Subscriber, 65
- subscriber
 - com::toc::coredx::DDS::rpc::ClientParams, 233
 - com::toc::coredx::DDS::rpc::ReplierParams, 236
 - com::toc::coredx::DDS::rpc::RequesterParams, 243, 244
 - com::toc::coredx::DDS::rpc::ServiceParams, 255
- SubscriberListener, 162
- SubscriberQos, 100
- SubscriptionBuiltinTopicData, 190
- SubscriptionBuiltinTopicDataDataReader, 189
- SubscriptionMatchedStatus, 175
- suspend_publications
 - com::toc::coredx::DDS::Publisher, 55
- TIMEBASEDFILTER_QOS_POLICY_ID
 - DDS Quality of Service, 8
- TOPIC_PRESENTATION_QOS
 - com::toc::coredx::DDS::PresentationQosPolicyAccessSupportKind, 145
- TOPIC_QOS_DEFAULT
 - DDS Quality of Service, 8
- TOPICDATA_QOS_POLICY_ID
 - DDS Quality of Service, 8
- TRANSIENT_DURABILITY_QOS
 - com::toc::coredx::DDS::DurabilityQosPolicyKind, 141
- TRANSIENT_LOCAL_DURABILITY_QOS
 - com::toc::coredx::DDS::DurabilityQosPolicyKind, 141
- TRANSPORTPRIORITY_QOS_POLICY_ID
 - DDS Quality of Service, 9
- TYPECODE_ALIAS
 - com::toc::coredx::DDS::TypeCodeKind, 147
- TYPECODE_ARRAY
 - com::toc::coredx::DDS::TypeCodeKind, 147
- TYPECODE_BOOLEAN
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_CHAR
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_DOUBLE
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_ENUM
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_FLOAT
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_LONGDOUBLE
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_LONGLONG
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_LONG
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_OCTET
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_SEQUENCE
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_SHORT
 - com::toc::coredx::DDS::TypeCodeKind, 148
- TYPECODE_STRING
 - com::toc::coredx::DDS::TypeCodeKind, 149
- TYPECODE_STRUCT
 - com::toc::coredx::DDS::TypeCodeKind, 149
- TYPECODE_ULONGLONG
 - com::toc::coredx::DDS::TypeCodeKind, 149
- TYPECODE_ULONG
 - com::toc::coredx::DDS::TypeCodeKind, 149
- TYPECODE_UNION
 - com::toc::coredx::DDS::TypeCodeKind, 149
- TYPECODE_USHORT
 - com::toc::coredx::DDS::TypeCodeKind, 149
- TYPECODE_VALUE
 - com::toc::coredx::DDS::TypeCodeKind, 149
- TYPECODE_WCHAR
 - com::toc::coredx::DDS::TypeCodeKind, 149
- TYPECODE_WSTRING
 - com::toc::coredx::DDS::TypeCodeKind, 149
- take
 - com::toc::coredx::DDS::FooDataReader, 81
- take_instance
 - com::toc::coredx::DDS::FooDataReader, 81
- take_next_instance
 - com::toc::coredx::DDS::FooDataReader, 82
- take_next_instance_w_condition
 - com::toc::coredx::DDS::FooDataReader, 82
- take_next_sample
 - com::toc::coredx::DDS::FooDataReader, 82
- take_replies
 - com::toc::coredx::DDS::rpc::Requester, 248
- take_reply
 - com::toc::coredx::DDS::rpc::Requester, 248, 249
- take_request
 - com::toc::coredx::DDS::rpc::Replier, 239
- take_requests
 - com::toc::coredx::DDS::rpc::Replier, 239
- take_w_condition
 - com::toc::coredx::DDS::FooDataReader, 82
- TcpTransport, 221
- TcpTransportConfig, 219
- thread_model
 - com::toc::coredx::DDS::DomainParticipantQos, 97
- ThreadModelQosPolicy, 131

- time_based_filter
 - com::toc::coredx::DDS::DataReaderQos, [91](#)
 - Time_t, [183](#)
 - TimeBasedFilterQosPolicy, [132](#)
 - Topic, [69](#)
 - topic_aliases
 - com::toc::coredx::DDS::RpcQosPolicy, [128](#)
 - topic_data
 - com::toc::coredx::DDS::TopicQos, [103](#)
 - TopicDataQosPolicy, [133](#)
 - TopicDescription, [71](#)
 - TopicListener, [164](#)
 - TopicQos, [102](#)
 - Transport, [222](#)
 - transport_priority
 - com::toc::coredx::DDS::DataWriterQos, [94](#)
 - com::toc::coredx::DDS::TopicQos, [104](#)
 - TransportPriorityQosPolicy, [134](#)
 - tx_max_packet_size
 - com::toc::coredx::DDS::TcpTransportConfig, [220](#)
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
 - tx_meta_multicast
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
 - tx_meta_unicast
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
 - type_consistency
 - com::toc::coredx::DDS::DataReaderQos, [91](#)
 - TypeCodeKind, [147](#)
 - TypeConsistencyEnforcementQosPolicy, [136](#)
 - TypeConsistencyKind, [150](#)
 - TypeSupport, [72](#)
 - TypecodeQosPolicy, [135](#)
 - ULongDynamicType, [211](#)
 - ULongLongDynamicType, [212](#)
 - USERDATA_QOS_POLICY_ID
 - DDS Quality of Service, [9](#)
 - UShortDynamicType, [214](#)
 - UdpTransport, [227](#)
 - UdpTransportConfig, [223](#)
 - unbind
 - com::toc::coredx::DDS::rpc::ServiceProxy, [257](#)
 - UnionDynamicType, [213](#)
 - unregister_instance
 - com::toc::coredx::DDS::FooDataWriter, [85](#)
 - unregister_instance_w_timestamp
 - com::toc::coredx::DDS::FooDataWriter, [85](#)
 - use_ipv4
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
 - use_ipv6
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
 - user_data
 - com::toc::coredx::DDS::DataReaderQos, [91](#)
 - com::toc::coredx::DDS::DataWriterQos, [94](#)
 - com::toc::coredx::DDS::DomainParticipantQos, [97](#)
 - user_multicast_address_v4
 - com::toc::coredx::DDS::UdpTransportConfig, [225](#)
 - user_multicast_address_v6
 - com::toc::coredx::DDS::UdpTransportConfig, [226](#)
 - UserDataQosPolicy, [137](#)
 - VOLATILE_DURABILITY_QOS
 - com::toc::coredx::DDS::DurabilityQosPolicyKind, [141](#)
 - valid_data
 - com::toc::coredx::DDS::SampleInfo, [62](#)
 - value
 - com::toc::coredx::DDS::DataRepresentationQosPolicy, [106](#)
 - com::toc::coredx::DDS::PeerParticipantQosPolicy, [122](#)
 - com::toc::coredx::DDS::Property_t, [182](#)
 - com::toc::coredx::DDS::PropertyQosPolicy, [124](#)
 - view_state
 - com::toc::coredx::DDS::SampleInfo, [62](#)
 - WRITERDATALIFECYCLE_QOS_POLICY_ID
 - DDS Quality of Service, [9](#)
 - wait
 - com::toc::coredx::DDS::WaitSet, [75](#)
 - wait_for_acknowledgments
 - com::toc::coredx::DDS::DataWriter, [31](#)
 - com::toc::coredx::DDS::Publisher, [55](#)
 - wait_for_historical_data
 - com::toc::coredx::DDS::DataReader, [28](#)
 - wait_for_replies
 - com::toc::coredx::DDS::rpc::Requester, [249](#)
 - wait_for_requests
 - com::toc::coredx::DDS::rpc::Replier, [239](#), [240](#)
 - wait_for_service
 - com::toc::coredx::DDS::rpc::ServiceProxy, [257](#), [258](#)
 - wait_for_services
 - com::toc::coredx::DDS::rpc::ServiceProxy, [258](#), [259](#)
 - WaitSet, [74](#)
 - com::toc::coredx::DDS::WaitSet, [74](#)
 - write
 - com::toc::coredx::DDS::FooDataWriter, [85](#)
 - write_w_timestamp
 - com::toc::coredx::DDS::FooDataWriter, [86](#)
 - writer_data_lifecycle
 - com::toc::coredx::DDS::DataWriterQos, [94](#)
 - WriterDataLifecycleQosPolicy, [138](#)
-