

RTI Connex Python API

Version 7.3.0.



Your systems.
Working as one.

Contents

1	Getting Started	2
2	Contents	3
2.1	API Overview	3
2.1.1	Hello World	3
2.1.2	DomainParticipant	4
2.1.3	Topics	5
	Special Topics	5
2.1.4	Publications	6
2.1.5	Subscriptions	6
	Creating a DataReader	6
	Reading data	7
	Being notified when data is available	7
	Special DataReaders	8
2.1.6	Data Types	9
	IDL to Python mapping	9
	Built-in types	16
	Type support	17
	DynamicType and DynamicData	18
2.1.7	XML Application Creation	20
	Defining your system in a configuration file	20
	Creating the Entities from the configuration	21
	Using IDL and Python data types	21
2.1.8	Request-Reply and Remote Procedure Calls	22
	Requesters and Repliers	22
	Remote Procedure Calls	25
2.2	Quick Reference	30
2.3	rti package	31
2.3.1	rti.connextdds	31
2.3.2	rti.types	759
	rti.types.builtin	759
2.3.3	rti.rpc	760
2.3.4	rti.asyncio	770
2.3.5	rti.logging	770
	Submodules	770
	Module contents	773
2.4	Examples	774

2.5	Copyrights, Notices, License	774
3	Additional documentation	776
	Python Module Index	777
	Index	778

RTI® Connex® is a connectivity framework for building distributed applications with requirements for high performance and scalability.

The **RTI Connex Python API** provides access to most Connex features from Python. It is supported on Python 3.6 to 3.12, and on Linux® x64 and arm64, Windows® x64, and macOS® x64 and arm64.

Chapter 1

Getting Started

The [RTI Connex Getting Started Guide](#) helps you install the software and run your first RTI Connex Python application while learning general concepts of Connex.

To learn more about the Python API, see *API Overview*; you can start with the *Hello World* example.

If you're looking for specific documentation, check out the *Quick Reference*, or the full *API reference*.

If you're migrating from *RTI Connector*, the [Connex Migration Guide](#) on [RTI Community](#) provides guidance.

Chapter 2

Contents

2.1 API Overview

2.1.1 Hello World

This section shows the basic code to publish and subscribe to a topic defined by a simple data type. Similar code can be generated by *rtiddsgen* from an IDL or XML file defining your data types. The sections that follow expand on the concepts shown in this example.

Define your types:

```
# hello.py

import rti.types as idl

@idl.struct
class HelloWorld:
    message: str = ""
```

Create a *DataWriter* to publish the HelloWorld Topic:

```
# hello_publisher.py

import time
import rti.connextdds as dds
from hello import HelloWorld

participant = dds.DomainParticipant(domain_id=0)
topic = dds.Topic(participant, 'HelloWorld Topic', HelloWorld)
writer = dds.DataWriter(participant.implicit_publisher, topic)

for i in range(10):
    writer.write(HelloWorld(message=f'Hello World! #{i}'))
    time.sleep(1)
```

Create a *DataReader* to subscribe to the HelloWorld Topic:

```
# hello_subscriber.py

import rti.connextdds as dds
import rti.asyncio
from hello import HelloWorld

participant = dds.DomainParticipant(domain_id=0)
topic = dds.Topic(participant, 'HelloWorld Topic', HelloWorld)
reader = dds.DataReader(participant.implicit_subscriber, topic)

async def print_data():
    async for data in reader.take_data_async():
        print(f"Received: {data}")

rti.asyncio.run(print_data())
```

Continue to the next sections to learn about a *DomainParticipant*, *Topics*, *Publications*, *Subscriptions*, and *Data Types*.

2.1.2 DomainParticipant

DomainParticipants are the focal point for creating, destroying, and managing other *Connex* objects. A DDS domain is a logical network of applications: only applications that belong to the same DDS domain may communicate using *Connex*. A DDS domain is identified by a unique integer value known as a domain ID. An application participates in a DDS domain by creating a *DomainParticipant* for that domain ID.

The following code creates a *DomainParticipant* on domain 0.

```
import rti.connextdds as dds
participant = dds.DomainParticipant(domain_id=0)
```

Like all *IEntity* types, *DomainParticipants* have QoS policies and listeners. The following example shows how to create a *DomainParticipant* with a specific QoS policy:

```
qos = dds.DomainParticipantQos()
qos.database.shutdown_cleanup_period = dds.Duration.from_milliseconds(10)
participant = dds.DomainParticipant(domain_id=0, qos=qos)
```

A *DomainParticipant* and its contained entities can also be created from an XML definition with the *QosProvider.create_participant_from_config()* function.

DomainParticipants (and all other *Entities*) get destroyed automatically when they are garbage collected; however, to ensure that they are destroyed at a certain point in your application, you can call `close()` or create them within a `with` block:

```
with dds.DomainParticipant(domain_id=0) as participant:
    print(participant.domain_id)
    # ...
```

2.1.3 Topics

Shared knowledge of the data types is a requirement for different applications to communicate with DDS. The applications must also share a way to identify what data is to be shared. Data (regardless of type) is uniquely distinguished by using a name called a *Topic*.

Topics allow for *Publishers* and *Subscribers* to communicate without knowing about each other. They dynamically discover each other through *Topics*. The data that the *DataReader* and *DataWriter* share is described by a *Topic*.

Topics are identified by a name, and they are associated with a type and a *DomainParticipant*.

The following code creates a *Topic* named “Car Position” for a type *Point*:

```
topic = dds.Topic(participant, "Car Position", Point)
```

Where *Point* can be defined in IDL as:

```
struct Point {
    int64 x;
    int64 y;
};
```

And in Python as follows:

```
import rti.types as idl

@idl.struct
class Point:
    x: int = 0
    y: int = 0
```

Data Types explains how to define your types in more detail.

Special Topics

In addition to the class *Topic*, there are a few separate *Topic* classes for certain types:

- For DynamicData *Topics*: *DynamicData.Topic* (see *DynamicType* and *DynamicData*)
- For the built-in discovery *Topics*: *ParticipantBuiltinTopicData.Topic*, *SubscriptionBuiltinTopicData.Topic*, *PublicationBuiltinTopicData.Topic*, *TopicBuiltinTopicData.Topic*

2.1.4 Publications

An application uses *DataWriters* to send data. A *DataWriter* is associated with a single *Topic*. You can have multiple *DataWriters* and *Topics* in a single application. In addition, you can have more than one *DataWriter* for a particular *Topic* in a single application. *Publishers* own and manage *DataWriters*.

To create a *DataWriter*, you need a *Topic* (see *Topics*) and a *DomainParticipant*. Additionally, you may add a QoS parameter and a listener.

The following code creates a *DataWriter* for the *Topic* we created in the *Topics* section:

```
publisher = dds.Publisher(participant)
writer = dds.DataWriter(publisher, topic)
```

To publish data, create a sample, set the values, and write it:

```
data = Point(x=1, y=2)
writer.write(data)
```

A special *DataWriter* type for *DynamicData*, *DynamicData.DataWriter* is also available. Find more information in *DynamicType* and *DynamicData*.

2.1.5 Subscriptions

An application uses *DataReaders* to access data received over DDS. A *DataReader* is associated with a single *Topic*. You can have multiple *DataReaders* and *Topics* in a single application. In addition, you can have more than one *DataReader* for a particular *Topic* in a single application. *Subscribers* own and manage *DataReaders*.

Creating a DataReader

To create a *DataReader*, you need to create a *Topic* (see *Topics*) and a *DomainParticipant*. Optionally, you can add a QoS parameter and a listener.

The following code creates a *DataReader* for the *Topic* we created in the *Topics* section:

```
subscriber = dds.Subscriber(participant)
reader = dds.DataReader(subscriber, topic)
```

A *DataReader* can be created with a *ContentFilteredTopic*, instead of a regular *Topic* to define a content-based subscription with a filter on the data type.

Reading data

When data is available, the `DataReader.read_data()` and `DataReader.take_data()` methods return a collection of data samples of the type specified in the *Topic*. (`take_data` removes the data from the reader, and `read_data` keeps it so it can be accessed again.)

```
for sample in reader.take_data():
    print(sample)
```

To also access the meta-data associated with each data sample, use `DataReader.take()` or `DataReader.read()`:

```
for data, info in reader.take():
    if info.valid:
        print(f"Data received: {data}")
    else:
        print(f"State changed: {info.state}")
```

Each call to these methods returns newly created objects (even `read()`), unlike other Connex language bindings, which return temporary loaned objects.

The `DataReader.select()` method allows selecting which data to read.

Being notified when data is available

There are a few ways to check if a reader has data available:

- Polling for data
- Using a Condition and a WaitSet
- Reading with an asynchronous generator
- Using the DataReaderListener

Polling for data means that you call the “read” or “take” methods described before at certain intervals to check if they return any data.

A `StatusCondition` and a `WaitSet` allows waiting synchronously until a `DataReader` status change triggers, including the `DATA_AVAILABLE` status:

```
def process_data(_):
    nonlocal reader
    for sample in reader.take_data():
        print(sample)

# Each Entity has a StatusCondition
status_condition = dds.StatusCondition(reader)

# Specify which status to get notified about and set the handler:
status_condition.enable_statuses = dds.StatusMask.DATA_AVAILABLE
status_condition.handler(process_data)
```

```
# Attach the condition to a waitset and call dispatch() to execute the
# condition handlers when they become active
waitset = dds.WaitSet()
waitset += status_condition
while True:
    waitset.dispatch(dds.Duration(4)) # Wait up to 4 seconds
```

The `async` versions of the “take” methods provide a simple way to write your subscriber application. The methods `DataReader.take_data_async()` or `DataReader.take_async()` work as asynchronous generators, returning data as it is received and awaiting as necessary.

To use these functions your application must import `rti.asyncio`, which requires Python 3.7+.

```
import rti.connextdds as dds
import rti.asyncio

async def print_infinite(reader: dds.DataReader):
    # Print data as it arrives, suspending the coroutine until data is
    # available.
    async for data in reader.take_data_async():
        print(data)

if __name__ == "__main__":
    # ... create the participant, topic, and subscriber as shown before
    reader = dds.DataReader(subscriber, topic)

    # you can use Python's asyncio.run() as well
    rti.asyncio.run(print_infinite(reader))
```

`take_data_async()` and `take_async()` receive an optional condition argument (a `dds.ReadCondition` or `dds.QueryCondition`) that can select data by state or content.

Finally, you can use a `DataReaderListener` to get notified of status updates, including new data. This method is only recommended for lightweight processing, since the listener callback is executed in an internal Connex thread, and should not block or perform CPU-heavy operations.

Special DataReaders

In addition to the class `DataReader`, there are a few separate

`DataReader` classes for certain types:

- For DynamicData *DataReaders*: `DataReader.Topic` (see *DynamicType and DynamicData*)
- For the built-in discovery *Topics*: `ParticipantBuiltinTopicData.DataReader`,
`SubscriptionBuiltinTopicData.DataReader`,
`PublicationBuiltinTopicData.DataReader`, `TopicBuiltinTopicData.DataReader`

2.1.6 Data Types

IDL to Python mapping

If you're not familiar with IDL and DDS data types, you can read and complete the exercises in the [RTI Connex Getting Started Guide](#), in the chapter on Data Types.

You can generate your Python data types from IDL (or XML) using **rtiddsgen**:

```
rtiddsgen -language python MyTypes.idl [-example universal]
```

(The optional `-example universal` option generates example publisher and subscriber scripts.)

You can also define your types directly in Python, without code generation, following a few conventions.

IDL structs map to Python dataclasses with a special decorator. For example, the following IDL struct:

```
struct Point {
    int64 x;
    int64 y;
};
```

Is defined in Python as follows:

```
import rti.types as idl

@idl.struct
class Point:
    x: int = 0
    y: int = 0
```

The `@idl.struct` decorator parses the type and generates the necessary type support routines that allow DDS to serialize and deserialize instances of this class. `@idl.struct` also applies `@dataclass` to the class, adding all its functionality (`__init__`, `__eq__`, `__repr__`, `__hash__`, `copy.deepcopy()` support, etc.). For example:

```
point1 = Point(x=1, y=2)
point2 = Point(y=2) # x=0, y=2
point3 = Point() # x=0, y=0

print(point1)
assert point1 != point2

point2.x = 1
assert point1 == point2

point4 = copy.deepcopy(point3)
assert point4 == point3
```

Point can be used to create a *Topic* as we explained in *Topics*.

There are a few pre-defined classes available for convenience (see *Built-in types* below).

The following sections explain how different IDL types map to Python.

Primitive types

The IDL types `int64`, `double`, and `bool` map directly to the Python built-in types `int`, `float`, and `bool`.

For IDL types of different sign or size, type annotations are provided. For example, the IDL types `int32`, `uint16`, and `float` map to the type annotations `idl.int32`, `idl.uint16`, and `idl.float32`, respectively.

For example, the following IDL type:

```
struct Foo {
    int32 a;
    int64 b;
    uint64 c;
    float d;
    double e;
    bool f;
};
```

Maps to the following Python class:

```
@idl.struct
class Foo:
    a: idl.int32
    b: int
    c: idl.uint64
    d: idl.float32
    e: float
    f: bool
```

(Note that IDL's `double` maps to Python's `float`, and IDL's `float` to `idl.float32`.)

Warning: The sign or size of the types is currently not enforced. If you write a value outside the expected range, the subscribers will receive an incorrect value.

Strings

IDL strings map to Python's built-in `str`.

IDL strings can be single (UTF-8) or wide (UTF-16) and bounded or unbounded. These options are passed to the `member_annotations` argument of the type decorator, if needed. By default strings are UTF-8 and unbounded.

For example, the following IDL type:

```
// MyTypes.idl
struct MyStrings {
    string unbounded_str;
    string<128> bounded_str;
```

```
wstring<256> bounded_wstr;
};
```

Maps to the following Python dataclass:

```
@idl.struct(
    member_annotations = {
        'bounded_str': [idl.bound(128)],
        'bounded_wstr': [idl.bound(256), idl.utf16],
    }
)
class MyStrings:
    unbounded_str: str = ""
    bounded_str: str = ""
    bounded_wstr: str = ""
```

The type above can be generated with **rtiddsgen** as follows:

```
rtiddsgen -unboundedSupport -language python MyTypes.idl
```

Sequences

The mapping of IDL sequences depends on whether the element type is a primitive type or not.

Non-primitive sequences map to Python's `list`.

Primitive sequences map, by default, to efficient, compact collections in the `dds` module. For example, an IDL sequence<int32> maps to `dds.Int32Seq`.

IDL sequences can be bounded or unbounded. Bounded sequences may not exceed the number of elements indicated by the bound when the data is written. The bound is specified as part of the `member_annotations` argument to the type decorator.

For example, the following IDL type:

```
struct MySequences {
    sequence<int64, 100> bounded_int64_seq;
    sequence<uint32> unbounded_uint32_seq;
    sequence<Foo> unbounded_foo_seq;
};
```

Maps to the following Python type:

```
@idl.struct(
    member_annotations = {
        'bounded_int64_seq': [idl.bound(100)],
    }
)
class MySequences:
    bounded_int64_seq: Sequence[int] = field(default_factory = idl.array_
↪factory(int))
    unbounded_uint32_seq: Sequence[idl.uint32] = field(default_factory = idl.
```

```
↪array_factory(idl.uint32)
    unbounded_foo_seq: Sequence[Foo] = field(default_factory = list)
```

The `field` function and its `default_factory` argument indicate how the dataclass is created by default. When a instance of `MySequences` is created, all the sequences are empty. You can add elements or replace them altogether. For example:

```
my_sequences = MySequences()
my_sequences.bounded_int64_seq.append(1)
my_sequences.bounded_int64_seq.append(2)
my_sequences.unbounded_uint32_seq = dds.Uint32Seq([33] * 5)
my_sequences.unbounded_foo_seq = [Foo(a=x) for x in range(10)]
```

You're not restricted to `dds.Int64Seq` or `dds.Uint32Seq`; you can write a `list`, but the data serialization will be less efficient.

Arrays

The mapping for IDL arrays is similar to that of sequences, except that an array must always have the same number of elements.

The default creation of data samples with arrays populates them with the right number of elements.

The `write()` operation will fail if a sample with an array containing an incorrect number of elements is written.

Warning: Multi-dimensional arrays are not fully supported in this release. They are flattened out and the number of elements is the product of the array's dimensions.

Nested collections

In IDL, you can define sequences of sequences, sequences of arrays, and arrays of sequences.

To do that, the inner collection must be aliased. For example:

```
typedef sequence<Point> PointSeq;
typedef int64 TenInts[10];

struct MySequences {
    sequence<PointSeq> sequence_of_point_sequences;
    sequence<TenInts> sequence_of_int_arrays;
    PointSeq five_point_sequences[5];
};
```

Optional members

By default, members of an IDL `struct` always contain a value and they are always published with a data sample. A member can be declared optional in IDL allowing it to not be sent with all data samples.

In Python, optional members receive the `None` value by default.

For example, the following IDL struct contains a required and an optional member:

```
struct MyOptionals {
    double required_value;
    @optional double optional_value;
};
```

This maps to the following Python dataclass:

```
@idl.struct
class MyOptionals:
    required_value: float = 0.0
    optional_value: Optional[float] = None
```

And a data sample is created by default as follows:

```
sample = MyOptionals()
assert sample.required_value == 0.0
assert sample.optional_value is None
```

Enumerations

IDL enumerations map to Python `IntEnum`-derived classes that are decorated with the `idl.enum` decorator.

```
enum Color {
    RED,
    GREEN,
    BLUE
};
```

Maps to:

```
@idl.enum
class Color(IntEnum):
    RED = 0
    GREEN = 1
    BLUE = 2
```


Unions

IDL unions define types in which only one member exists at a time. The selected member is identified by the “discriminator.”

IDL unions map to decorated Python dataclasses with two members (`discriminator` and `value`) and one read/write property for each member that allows setting the value and the discriminator consistently.

For example, the following IDL union:

```
union MyUnion switch(int32) {
    case 0:
        string string_member;
    case 1:
        int64 int_member;
    case 2:
        Point point_member;
};
```

Maps to the following Python class:

```
@idl.union
class MyUnion:

    discriminator: idl.int32 = 0
    value: Union[str, int, Point] = ""

    string_member: str = idl.case(0)
    int_member: int = idl.case(1)
    point_member: Point = idl.case(2)
```

The `discriminator` and `value` members should be used as read-only. To modify the union, use the “cases” (read/write properties). For example:

```
sample = MyUnion()

# By default the case with the lowest discriminator value (0 in this case)
# is selected (unless a "default:" label is defined in IDL)
assert sample.discriminator == 0
assert sample.value == ""
assert sample.string_member == ""

# Select a different member:
sample.point_member = Point(1, 2)
assert sample.discriminator == 2
assert sample.value == Point(1, 2)
assert sample.point_member == Point(1, 2)

# Attempting to access member that is not selected raises a ValueError:
try:
    print(sample.string_member)
except ValueError:
    print("string_member is not selected")
```

Modules

Each IDL (or XML) file called **Foo.idl** generates a Python file with the same name, **Foo.py**.

This defines a python package you can import:

```
import Foo

my_type = Foo.MyType()
```

Additionally, in IDL you can define “modules.” Similarly to C++ namespaces, an IDL module can be partially defined in several files. To allow for this capability, IDL modules map to Python’s [SimpleNamespace](#).

For example, assume the following IDL files:

```
# Foo.idl

module A {
    struct MyType1 { ... };
};

struct MyType2 { ... };
```

And:

```
# Bar.idl

module A {
    struct MyType3 { ... };
};

module B {
    struct MyType4 { ... };
};
```

This generates two Python packages, **Foo.py** and **Bar.py**. The module **A** is accessible from both packages as `Foo.A` and `Bar.A`. **Foo.idl** also defines a type without a module, and **Bar.idl** defines another module, **B**:

```
import Foo
import Bar

sample1 = Foo.A.MyType1()
sample2 = Foo.MyType2()
sample3 = Bar.A.MyType3()
sample4 = Bar.B.MyType4()

# You can create an alias:
MyType3 = Bar.A.MyType3

sample3 = MyType3()
```

IDL annotations

There are several IDL annotations that are passed to the `struct`, `union`, or `enum` decorators in the `type_annotations` or `member_annotations` arguments.

Examples are the `@key` and extensibility annotations (such as `@mutable`):

```
@mutable
struct MutableKeyedType {
    @key string id;
    string value;
};
```

The Python mapping is:

```
@idl.struct(
    type_annotations = [idl.mutable],
    member_annotations = {
        'id': [idl.key]
    }
)
class MutableKeyedType:
    id: str = ""
    value: str = ""
```

These annotations don't have a direct effect on how you use the classes in your application, but they may change how the data is internally processed or delivered.

Built-in types

For convenience, the following types are directly available in the `rti.types.builtin` package:

```
@idl.struct
class String:
    value: str = ""

@idl.struct(member_annotations={'key': [idl.key]})
class KeyedString:
    key: str = ""
    value: str = ""

@idl.struct
class Bytes:
    value: Sequence[idl.uint8] = field(default_factory=idl.array_factory(idl.
    ↪uint8))

@idl.struct(member_annotations={'key': [idl.key]})
class KeyedBytes:
    key: str = ""
```

```
value: Sequence[idl.uint8] = field(default_factory=idl.array_factory(idl.
→uint8))
```

You can directly use these types in your application:

```
import rti.connextdds as dds
from rti.types.builtin import String

participant = dds.DomainParticipant(domain_id=0)
topic = dds.Topic(participant, "HelloWorld", String)
writer = dds.DataWriter(participant, topic)
writer.write(String("Hello World!"))
```

Type support

Every `@idl.struct`-decorated class or `@idl.union`-decorated class has an associated `TypeSupport` object that can be obtained as follows:

```
import rti.types as idl

@idl.struct
class Foo:
    ...

foo_support = idl.get_type_support(Foo)
```

`TypeSupport` provides access to serialization functions:

```
foo = Foo()
buffer = foo_support.serialize(foo)
new_foo = foo_support.deserialize(buffer)
assert foo == new_foo
```

It also provides the property `max_serialized_sample_size`, and the method `get_serialized_sample_size()`.

`TypeSupport` allows converting data samples to and from JSON strings:

```
point_support = idl.get_type_support(Point)
point = Point(x=10, y=20)
assert point == point_support.from_json('{"x": 10, "y": 20}')
assert point_support.to_json(point) == '{"x": 10, "y": 20}'
```

`TypeSupport` also provides information about the type definition as a `DynamicType` (`dynamic_type` property) and helpers to convert to and from `DynamicData` (`to_dynamic_data()` and `from_dynamic_data()` methods).

DynamicType and DynamicData

The *Connext* Python API can dynamically load type definitions from XML and create `dds.DynamicData` samples.

```
import rti.connextdds as dds

provider = dds.QosProvider("your_types.xml")
my_type = provider.type("MyType")
```

You can now use `my_type` to create a `DynamicData.Topic` and to instantiate a `DynamicData` object:

```
topic = dds.DynamicData.Topic(participant, "Example MyType", my_type)
sample = dds.DynamicData(my_type)
sample["x"] = 42 # assuming MyType has an int32 field
```

You can use `topic` to create a `DynamicData.DataWriter` or a `DynamicData.DataReader`.

Types can also be defined dynamically in the application, using `DynamicType` and its derived classes.

The following example creates a type and instantiates a data sample:

```
# struct Point {
#     double x, y;
# };
point_type = dds.StructType("Point")
point_type.add_member(dds.Member("x", dds.Float64Type()))
point_type.add_member(dds.Member("y", dds.Float64Type()))

# struct MyType {
#     @key string<128> id;
#     Point location;
#     int32 int_array[5];
#     sequence<Point, 10> path;
# };
my_type = dds.StructType("MyType")
my_type.add_member(dds.Member(name="id", data_type=dds.StringType(128), is_
↪key=True))
my_type.add_member(dds.Member(name="location", data_type=point_type))
my_type.add_member(dds.Member(name="int_array", data_type=dds.ArrayType(dds.
↪Int32Type(), 5)))
my_type.add_member(dds.Member(name="path", data_type=dds.SequenceType(point_
↪type, 10)))

# Instantiate the type
sample = dds.DynamicData(my_type)
sample["id"] = "object1"
```

Accessing Nested Members

There are a few different ways to manipulate data with nested types. The `.` notation allows accessing nested primitive members at any level:

```
sample = dds.DynamicData(my_type)
sample["location.x"] = 1.5
sample["location.y"] = 2.5
```

To make multiple modifications to a complex member, you can get a temporary reference (a loan) to the member:

```
with sample.loan_value("location") as location:
    location.data["x"] = 11.5
    location.data["y"] = 12.5
```

A nested member can be assigned from a dictionary, too:

```
sample["location"] = {"x": 4.5, "y": 5.5}
print(sample["location"])
```

Accessing Sequences and Arrays

Sequences and arrays can be retrieved or set using Python lists:

```
# We're using the type we created before
sample = dds.DynamicData(my_type)

# Set the array field with the values of a python list
sample["int_array"] = [1, 2, 3, 4, 5]

# Get all the array elements in a python list
lst = list(sample["int_array"])

# Set and get a single element:
sample["int_array[1]"] = 4
value = sample["int_array[1]"]
```

Lists of structures can be accessed using lists of dictionaries:

```
sample["path"] = [{"x": 1, "y": 2}, {"x": 3, "y": 4}]
path = list(sample["path"])
```

If you only need to set a few elements or fields, you can loan the sequence and its elements. Sequences are automatically resized when you access and index above the current length:

```
with sample.loan_value("path") as path:
    with path.data.loan_value(2) as point:
        point.data["x"] = 111
        point.data["y"] = 222
print(sample["path[2].x"]) # prints 111
```

2.1.7 XML Application Creation

RTI Connex supports the use of XML for the complete system definition. This includes not only the definition of the data types and Quality of Service settings, but also the definition of the Topics, DomainParticipants, and all the Entities they contain (Publishers, Subscribers, DataWriters and DataReaders).

The application calls `QosProvider.create_participant_from_config()` to indicate the participant configuration name of the DomainParticipant that the application wants to create. This method takes care of the rest: creating the DomainParticipant, registering the types and creating Topics, and populating all the configured Entities. When the application needs to read or write data, register listeners, or perform any other action, it simply looks up the appropriate Entity by name and uses it.

Defining your system in a configuration file

The following example configures types and DDS Entities in XML:

```
<!-- my_dds_system.xml -->
<dds>
  <types>
    <struct name="Foo">
      <member name="x" type="int32"/>
    </struct>
  </types>

  <domain_library name="ExampleDomainLibrary" >
    <domain name="ExampleDomain" domain_id="0">
      <register_type name="Foo" type_ref="Foo"/>
      <topic name="ExampleTopic" register_type_ref="Foo"/>
    </domain>
  </domain_library>

  <domain_participant_library name="ExampleParticipantLibrary">
    <domain_participant name="ExamplePublicationParticipant" domain_ref=
    ↪ "ExampleDomainLibrary::ExampleDomain">
      <publisher name="ExamplePublisher">
        <data_writer name="ExampleWriter" topic_ref="ExampleTopic"/>
      </publisher>
    </domain_participant>
    <domain_participant name="ExampleSubscriptionParticipant" domain_ref=
    ↪ "ExampleDomainLibrary::ExampleDomain">
      <data_reader name="ExampleReader" topic_ref="ExampleTopic"/>
    </domain_participant>
  </domain_participant_library>
</dds>
```

See the [XML-Based Application Creation Getting Started Guide](#) for more information.

Creating the Entities from the configuration

To load one of the DomainParticipants defined above, use the following code:

```
qos_provider = dds.QosProvider("my_dds_system.xml")
participant = qos_provider.create_participant_from_config(
    "ExampleParticipantLibrary::ExamplePublicationParticipant"
)
```

This creates all the participant's contained entities as well.

The type in this example was also defined in XML, and therefore the DataReaders and DataWriters that use it are *DynamicData.DataReader* and *DynamicData.DataWriter*. The following code looks up the DataWriter, creates a data sample, and writes it:

```
writer = dds.DynamicData.DataWriter(
    participant.find_datawriter("ExamplePublisher::ExampleWriter")
)
sample = writer.create_data()
sample["x"] = 10
writer.write(sample)
```

You can look up the DataReader with *DomainParticipant.find_datareader()*.

Using IDL and Python data types

You can also define your types in IDL and Python (as described in *Data Types*) and refer to them in the XML configuration. To use a Python type (whether it is defined directly in Python or generated from IDL), the application must previously register it with the same name used in the XML configuration file.

For example, given the following Python type:

```
@idl.struct
class Point:
    x: int = 0
    y: int = 0
```

We can rewrite the `<domain_library>` in the previous XML configuration to refer to `Point` as follows:

```
<domain_library name="ExampleDomainLibrary" >
  <domain name="ExampleDomain" domain_id="0">
    <register_type name="Point"/>
    <topic name="ExampleTopic" register_type_ref="Point"/>
  </domain>
</domain_library>
```

And then register the type before creating the participant:

```
dds.DomainParticipant.register_idl_type(Point, "Point")

qos_provider = dds.QosProvider("my_dds_system.xml")
```



```
participant = qos_provider.create_participant_from_config(
    ↪"ExamplePublicationParticipant")
```

Now the writer writes `Point` objects instead of `DynamicData`:

```
writer = dds.DataWriter(
    participant.find_datawriter("ExamplePublisher::ExampleWriter")
)
writer.write(Point(x=10, y=20))
```

2.1.8 Request-Reply and Remote Procedure Calls

As applications become more complex, it often becomes necessary to use other communication patterns in addition to publish-subscribe. Sometimes an application needs to get a one-time snapshot of information; for example, to make a query into a database or retrieve configuration parameters that never change. Other times an application needs to ask a remote application to perform an action on its behalf; for example, to invoke a remote procedure call or a service.

The request-reply pattern has two roles: the `rti.rpc.Requester` (service consumer or client) sends a request message and waits for a reply message. The `rti.rpc.Replier` (service provider) receives the request message and responds with a reply message.

The Remote Procedure Call pattern is a specialization of the Request-Reply one. In this pattern a client and a server communicate using a functional interface. The client application calls functions on the interface, and the server application implements the functions.

The Connex Core Libraries User's Manual provides more information about these communication patterns in [Alternative Communication Models](#).

A full request-reply example application is available in the [Connex Examples repository](#).

Requesters and Repliers

To create a *Requester* or a *Replier*, you need the following:

- A *DomainParticipant* (see *DomainParticipant*)
- The *request type* and the *reply type* (see *Data Types*)
- A *service name*, or alternatively, you can specify the *request topic name* and the *reply topic name*.

A *Requester* and a *Replier* will communicate when they join the same *domain id* and use the same *service name* (or same topic names), compatible *request* and *reply* types, and compatible *QoS*.

The following code creates a `rti.rpc.Requester`:

```
import rti.connextdds as dds
from rti.rpc import Requester

participant = dds.DomainParticipant(domain_id=0)
```

```
requester = Requester(
    request_type=MyRequestType,
    reply_type=MyReplyType,
    participant=participant,
    service_name="MyServiceName"
)
```

The same or a different application can create a `rti.rpc.Replier` as follows:

```
from rti.rpc import Replier

replier = Replier(
    request_type=MyRequestType,
    reply_type=MyReplyType,
    participant=participant,
    service_name="MyServiceName"
)
```

For this example `MyRequestType` and `MyReplyType` are defined as follows:

```
import rti.types as idl

@idl.struct
class MyRequestType:
    x: int = 0
    y: int = 0

@idl.struct
class MyReplyType:
    result: int = 0
```

The `request_type` or `reply_type` arguments can also be instances of `rti.connextdds.DynamicType` or one of the built-in types (`rti.types.builtin`).

There also are a number of optional parameters you can specify, including the quality of service.

Requester: sending requests and receiving replies

To send a request, create an instance of the request type and send it:

```
request = MyRequestType(x=1, y=2)
requester.send_request(request)
```

After sending a request, the Requester will typically wait for one or more replies:

```
replies = requester.receive_replies(dds.Duration(seconds=20))
for data, info in replies:
    print(data)
```

The `receive_replies` operation is equivalent to `wait_for_replies` followed by `take_replies`.

Replier: receiving requests and sending replies

The replier will typically process requests in a loop. The following example receives requests, processes each of them to produce a reply, and sends it.

```
while True:
    requests = replier.receive_requests(dds.Duration(seconds=20))
    for data, info in requests:
        reply = MyReplyType(result=data.x + data.y)
        replier.send_reply(reply, info)
```

Note how `send_reply` expects the `SampleInfo` from the request. This will allow delivering the reply only to the requester that sent the request. Using the same `info` object, multiple replies can be sent for the same request; and replies can be sent in any order: the `data` and `info` objects can be saved for later use.

SimpleReplier

The `rti.rpc.SimpleReplier` class is a convenience class that allows for an easy implementation of the replier application using a callback function.

The following example shows how to create a `SimpleReplier` that implements the same behavior as the `Replier` above:

```
from rti.rpc import SimpleReplier

simple_replier = SimpleReplier(
    request_type=MyRequestType,
    reply_type=MyReplyType,
    participant=participant,
    service_name="MyServiceName",
    handler=lambda request: MyReplyType(result=request.x + request.y)
)
```

The `handler` argument is a function that receives a single argument of the type `request_type` and must return an instance of the type `reply_type`. A `SimpleReplier` is only suitable for applications where each request generates exactly one reply and when the process to generate the reply is fast (otherwise the handler function would hold internal Connex threads).

Waiting for a reply to a specific request

A *Requester* can send multiple requests and then wait for replies to any of them or to a specific one.

When sending requests, save the returned request id:

```
request_id1 = requester.send_request(MyRequestType(x=1, y=1))
request_id2 = requester.send_request(MyRequestType(x=2, y=2))
```

Then, when waiting for and taking the replies, you can specify the request id:

```

if not requester.wait_for_replies(
    related_request_id=request_id2,
    max_wait=dds.Duration(seconds=20)):

    raise Exception("Timeout waiting for replies")

replies = requester.take_replies(related_request_id=request_id2)
for data, info in replies:
    print(data) # Prints MyReplyType(result=4)

```

The first operation, `wait_for_replies`, will wait until at least one reply for `request_id2` is received. The second operation, `take_replies`, will take all replies for `request_id2`.

Accessing the underlying readers and writers

Requesters and *Repliers* each contain one *DataWriter* and one *DataReader* for the request and reply topics. These can be accessed for advanced use cases.

For example, the *Replier's* reader can be used to asynchronously read requests, which allows simplifying the replier loop as follows:

```

async def process_requests(replier: Replier):
    async for data, info in replier.request_datareader.take_async():
        reply = MyReplyType(result=data.x + data.y)
        replier.send_reply(reply, info)

```

For more information, see [Accessing Underlying DataWriters and DataReaders](#) in the Connex Core Libraries User's Manual.

Remote Procedure Calls

Warning: This feature is experimental and subject to change in future releases. It is included in this release to gather customer interest and feedback. For this reason, do not deploy any applications using Remote Procedure Calls in production. The Requester and Replier APIs described above can be used in production.

For support, you may contact support@rti.com.

With this pattern, a service interface allows a client to make remote function calls into a service. A service interface can be defined as follows:

```

import abc
import rti.types as idl
import rti.rpc as rpc

@idl.struct
class Coordinates:

```

```

x: int = 0
y: int = 0

@rpc.service
class RobotControl(abc.ABC):
    @rpc.operation
    async def walk_to(self, destination: Coordinates,
                     speed: idl.float32) -> Coordinates: ...

    @rpc.operation
    async def get_speed(self) -> idl.float32: ...

```

A service interface such as `RobotControl` is an abstract base class (ABC) decorated with the `@rpc.service` decorator and with a set of `async` abstract methods decorated with the `@rpc.operation` decorator. The names and types of each operation and its parameters are used to synthesize the DDS topic-types that allow the client and service to communicate. The parameters and return values of an operation can be other IDL-based types (see *Data Types*).

The Python class `RobotControl` is derived from this IDL definition:

```

struct Coordinates {
    int64 x;
    int64 y;
};

@service
interface RobotControl {
    Coordinates walk_to(Coordinates destination, float speed);
    float get_speed();
};

```

Warning: Code generation from IDL interface or exception definitions to Python is not supported in this release. Other features, such as out parameters or attributes are not supported either.

From the `RobotControl` class you can define a client and a service.

The client class is defined simply as:

```

class RobotControlClient(RobotControl, rpc.ClientBase): ...

```

The base class `rpc.ClientBase` provides the implementation of all the methods decorated with `@rpc.operation`. The `RobotControlClient` class doesn't have any implementation of its own.

A `RobotControlClient` can be instantiated in a `DomainParticipant` with a service name:

```

import rti.connextdds as dds

participant = dds.DomainParticipant(domain_id=0)
client = RobotControlClient(participant, "My RobotControl")

```

And then used to make remote calls:

```
result = await client.walk_to(destination=Coordinates(x=10, y=10), speed=50.0)
print(result)
```

The call to `walk_to` will send a request to the server and return a coroutine that will complete when the reply is received, providing the return value.

On the service side, an implementation of `RobotControl`, with the server-side logic must be implemented. For example, we can define `RobotControlExample` as follows:

```
class RobotControlExample(RobotControl):
    def __init__(self):
        self.position = Coordinates(0, 0)
        self.speed = 0.0

    async def walk_to(self, destination: Coordinates, speed: idl.float32) ->
↳Coordinates:
        if self.speed <= 0.0:
            return self.position # return previously known position

        if speed > 100.0:
            self.speed = 100 # maximum speed
        else:
            self.speed = speed

        # sleep for 0 to 10 seconds, depending on the speed
        await asyncio.sleep(10.0 - 0.1 * self.speed)

        self.speed = 0.0
        self.position = destination
        return destination

    def get_speed(self) -> idl.float32:
        return self.speed
```

The implementation of each method can be asynchronous or synchronous.

To start receiving function calls into `RobotControlExample` we need to create a `rpc.Service` with an instance of `RobotControlExample`, a `DomainParticipant` and a service name:

```
participant = dds.DomainParticipant(domain_id=0)
robot_control = RobotControlExample()
service = rpc.Service(robot_control, participant, "My RobotControl")
await service.run() # runs forever
```

While `service` is running it will receive calls from matching clients (those with the same service name and compatible interfaces and QoS), call The corresponding method in `robot_control` and send the result of the operation back to the client that called it.

Concurrency model

Both the service and the client provide `async` methods and therefore need to be run in an `asyncio` event loop.

All of the client methods derived from the service interface are `async`; for each operation (such as `walk_to`) the client uses a `Requester` to send a request and returns a coroutine that completes when the reply from the service is received, providing the return value.

The server uses the `asyncio` event loop and a `Replier` to run the following tasks when `run()` is called:

- A *read* task that receives requests using the `Replier`.
- One or more *process* tasks that call the methods in the service implementation and send the result back using the `Replier`.

If a method is asynchronous (for example `RobotControlExample.walk_to` in the example above), the *process* task will await for the method to complete. If the method is not asynchronous (`RobotControlExample.get_speed`), it should return a value immediately to not block the event loop.

When all the *process* tasks are busy, the *read* task will stop receiving requests. The number of *process* tasks can be configured with the `task_count` Service parameter:

```
server = rpc.Service(robot_control, participant, "My RobotControl",
                    task_count=10)
```

The default value is 4. Note that with a value of 1, the server will be forced to process requests sequentially in the order they are received.

The service's `run()` method runs until it is cancelled. For example:

```
service = rpc.Service(robot_control, participant, "My RobotControl")
await service.run() # runs forever

# or

service = rpc.Service(robot_control, participant, "My RobotControl")
service_task = asyncio.create_task(service.run())
await asyncio.sleep(10.0) # run for 10 seconds
service_task.cancel()
await service_task
```

Note that `asyncio` event loops are single-threaded. If your service implementation needs to run an operation in a separate thread, you can use the `asyncio.run_in_executor` function. For example, the following implementation of `walk_to` runs the synchronous function `move_robot_impl` in a separate thread:

```
class RobotControlExample(RobotControl):
    async def walk_to(self, destination: Coordinates, speed: idl.float32) ->_
    ->Coordinates:
        # ...
        self.speed = speed

        # run move_robot_impl in a separate thread
        loop = asyncio.get_running_loop()
```

```

await loop.run_in_executor(None, move_robot_impl, destination, speed)

self.speed = 0.0
self.position = destination
return destination

```

If this concurrency model doesn't fit your needs, you can use the `Requester` and `Replier` classes directly, which provide more flexibility.

Exceptions

Operations in a service interface may throw exceptions. Exceptions are `@idl.struct` classes that inherit from `Exception`. Operations need to declare all the exceptions that they may throw. When a service implementation throws an exception, the `rpc.Service` will catch it and send it to the client, which will rethrow it.

For example, to have the `walk_to` operation throw a `TooFast` exception when the speed is too high, we can define the exception as follows:

```

@idl.struct
class TooFast(Exception):
    speed: idl.float32

```

And declare it in the `walk_to` operation:

```

@rpc.service
class RobotControl:
    @rpc.operation(raises=[TooFast])
    async def walk_to(self, destination: Coordinates, speed: idl.float32) ->_
    ↪Coordinates: ...

```

We can now modify `ExampleRobotControl` to throw `TooFast`:

```

class RobotControlExample(RobotControl):
    async def walk_to(self, destination: Coordinates, speed: idl.float32) ->_
    ↪Coordinates:
        if speed > 100.0:
            raise TooFast(speed)

    # ...

```

The client can now catch the exception:

```

try:
    await client.walk_to(destination, speed)
except TooFast as e:
    print(f"Too fast: {e.speed}")

```

Exceptions that are not declared in the `raises` parameter of an operation will be propagated and raised in the client as `rpc.RemoteUnknownExceptionError`.

2.2 Quick Reference

This section includes links to common `rti.connexdds` types.

See also *API Overview*

DDS Entities:

- *DomainParticipant*
- *Publisher*
- *Subscriber*
- *Topic, ContentFilteredTopic*
- *DataReader*
- *DataWriter*

Data types:

- User types: *Data Types*
- *rti.types.builtin*
- *DynamicData*

Quality of Service (QoS):

- *QosProvider*
- *DomainParticipantQos*
- *TopicQos*
- *PublisherQos*
- *SubscriberQos*
- *DataReaderQos*
- *DataWriterQos*

Listeners:

- *DomainParticipantListener*
- *TopicListener*
- *PublisherListener*
- *SubscriberListener*
- *DataReaderListener*
- *DataWriterListener*

Conditions:

- *WaitSet*

- *Condition*
- *GuardCondition*
- *StatusCondition*
- *ReadCondition*
- *QueryCondition*

Full module documentation: *rti.connexdds*

2.3 rti package

The *rti* package contains the following packages and modules:

2.3.1 rti.connexdds

rti.connexdds (often aliased as *dds*) is the main package and contains the DDS API.

The following is the alphabetical list of all classes in the module. For a list of the most relevant ones, see *Quick Reference*.

class *rti.connexdds.ACTEnumMember*

Bases: *DynamicType*

INVALID_INDEX = 4294967295

__eq__ (*self*: *rti.connexdds.ACTEnumMember*, *arg0*: *rti.connexdds.ACTEnumMember*) → bool
Test for equality.

__getattr__ (*self*: *rti.connexdds.ACTEnumMember*, *arg0*: *str*) → *rti.connexdds.EnumMember*

__getitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__getitem__**(*self*: *rti.connexdds.ACTEnumMember*, *arg0*: *str*) -> *rti.connexdds.EnumMember*
2. **__getitem__**(*self*: *rti.connexdds.ACTEnumMember*, *arg0*: *int*) -> *rti.connexdds.EnumMember*

__hash__ = None

__init__ (**args*, ***kwargs*)

__module__ = '*rti.connexdds*'

__ne__ (*self*: *rti.connexdds.ACTEnumMember*, *arg0*: *rti.connexdds.ACTEnumMember*) → bool
Test for inequality.

cdr_serialized_sample_max_size (*self*: rti.connexdds.ACTEnumMember, *arg0*: int) → int

Gets the maximum serialized size of samples of this type.

cdr_serialized_sample_min_size (*self*: rti.connexdds.ACTEnumMember, *arg0*: int) → int

Gets the minimum serialized size of samples of this type.

extensibility_kind (*self*: rti.connexdds.ACTEnumMember) → *rti.connexdds.ExtensibilityKind*

Gets the extensibility kind.

find_member_by_name (*self*: rti.connexdds.ACTEnumMember, *arg0*: str) → int

Obtains the member index from its name.

member (**args*, ***kwargs*)

Overloaded function.

1. member(*self*: rti.connexdds.ACTEnumMember, *arg0*: int) -> rti.connexdds.EnumMember

Gets a member by its index.

2. member(*self*: rti.connexdds.ACTEnumMember, *arg0*: str) -> rti.connexdds.EnumMember

Gets a member by its name.

property member_count

Gets the number of members.

members (*self*: rti.connexdds.ACTEnumMember) → *rti.connexdds.EnumMemberSeq*

Gets a copy of all the members

class rti.connexdds.ACTMember

Bases: *DynamicType*

INVALID_INDEX = 4294967295

__eq__ (*self*: rti.connexdds.ACTMember, *arg0*: rti.connexdds.ACTMember) → bool

Test for equality.

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. __getitem__(*self*: rti.connexdds.ACTMember, *arg0*: str) -> rti.connexdds.Member

2. __getitem__(*self*: rti.connexdds.ACTMember, *arg0*: int) -> rti.connexdds.Member

__hash__ = None

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ACTMember, *arg0*: rti.connexdds.ACTMember) → bool
 Test for inequality.

cdr_serialized_sample_max_size (*self*: rti.connexdds.ACTMember, *arg0*: int) → int
 Gets the maximum serialized size of samples of this type.

cdr_serialized_sample_min_size (*self*: rti.connexdds.ACTMember, *arg0*: int) → int
 Gets the minimum serialized size of samples of this type.

extensibility_kind (*self*: rti.connexdds.ACTMember) → *rti.connexdds.ExtensibilityKind*
 Gets the extensibility kind.

find_member_by_name (*self*: rti.connexdds.ACTMember, *arg0*: str) → int
 Obtains the member index from its name.

member (*args, **kwargs)
 Overloaded function.

1. member(*self*: rti.connexdds.ACTMember, *arg0*: int) -> rti.connexdds.Member
 Gets a member by its index.
2. member(*self*: rti.connexdds.ACTMember, *arg0*: str) -> rti.connexdds.Member
 Gets a member by its name.

property member_count
 Gets the number of members.

members (*self*: rti.connexdds.ACTMember) → *rti.connexdds.MemberSeq*
 Gets a copy of all the members

class rti.connexdds.**ACTUnionMember**
 Bases: *DynamicType*

INVALID_INDEX = 4294967295

__eq__ (*self*: rti.connexdds.ACTUnionMember, *arg0*: rti.connexdds.ACTUnionMember) → bool
 Test for equality.

__getitem__ (*args, **kwargs)
 Overloaded function.

1. __getitem__(*self*: rti.connexdds.ACTUnionMember, *arg0*: str) -> rti.connexdds.UnionMember
2. __getitem__(*self*: rti.connexdds.ACTUnionMember, *arg0*: int) -> rti.connexdds.UnionMember

__hash__ = None

__init__ (*args, **kwargs)

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ACTUnionMember, *arg0*: rti.connexdds.ACTUnionMember) → bool
 Test for inequality.

cdr_serialized_sample_max_size (*self*: rti.connexdds.ACTUnionMember, *arg0*: int) → int

Gets the maximum serialized size of samples of this type.

cdr_serialized_sample_min_size (*self*: rti.connexdds.ACTUnionMember, *arg0*: int) → int

Gets the minimum serialized size of samples of this type.

extensibility_kind (*self*: rti.connexdds.ACTUnionMember) → *rti.connexdds.ExtensibilityKind*

Gets the extensibility kind.

find_member_by_name (*self*: rti.connexdds.ACTUnionMember, *arg0*: str) → int

Obtains the member index from its name.

member (**args*, ***kwargs*)

Overloaded function.

1. member(*self*: rti.connexdds.ACTUnionMember, *arg0*: int) -> rti.connexdds.UnionMember

Gets a member by its index.

2. member(*self*: rti.connexdds.ACTUnionMember, *arg0*: str) -> rti.connexdds.UnionMember

Gets a member by its name.

property member_count

Gets the number of members.

members (*self*: rti.connexdds.ACTUnionMember) → *rti.connexdds.UnionMemberSeq*

Gets a copy of all the members

class rti.connexdds.**AckResponseData**

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

Overloaded function.

1. __init__(*self*: rti.connexdds.AckResponseData) -> None

Create an empty ack response.

2. __init__(*self*: rti.connexdds.AckResponseData, sequence: rti.connexdds.Uint8Seq) -> None

Create an instance with a byte sequence.

__module__ = 'rti.connexdds'

property value

Gets the subscription handle of the acknowledging DataReader

```
class rti.connextdds.AcknowledgmentInfo
```

```
Bases: pybind11_object
```

```
__init__ (*args, **kwargs)
```

```
__module__ = 'rti.connextdds'
```

```
property cookie
```

Cookie value of the acknowledgment.

```
property response_data
```

User data payload of application-level acknowledgment.

```
property sample_identity
```

Get the identity of the sample being acknowledged.

```
property subscription_handle
```

Gets the subscription handle of the acknowledging DataReader

```
property valid_response_data
```

Flag indicating validity of the use response data in the acknowledgment.

```
class rti.connextdds.AcknowledgmentKind
```

```
Bases: pybind11_object
```

```
APPLICATION_AUTO = <AcknowledgmentKind.APPLICATION_AUTO: 1>
```

```
APPLICATION_EXPLICIT = <AcknowledgmentKind.APPLICATION_EXPLICIT: 3>
```

```
APPLICATION_ORDERED = <AcknowledgmentKind.APPLICATION_ORDERED: 2>
```

```
class AcknowledgmentKind
```

```
Bases: pybind11_object
```

Members:

PROTOCOL : Samples are acknowledged by RTPS protocol.

Samples are acknowledged according to the Real-Time Publish-Subscribe (RTPS) interoperability protocol.

APPLICATION_AUTO : Samples are acknowledged automatically after a subscribing application has accessed them.

The DataReader automatically acknowledges a sample after it has been taken and the loan returned.

APPLICATION_ORDERED : Samples up to a specified sequence number are acknowledged.

APPLICATION_EXPLICIT : Samples are acknowledged after the subscribing application explicitly calls `acknowledge` on the samples.

Samples received by a DataReader are explicitly acknowledged by the subscribing application, after it calls either `DataReader.acknowledge_all` or `DataReader.acknowledge_sample`.

```

APPLICATION_AUTO = <AcknowledgmentKind.APPLICATION_AUTO: 1>

APPLICATION_EXPLICIT =
<AcknowledgmentKind.APPLICATION_EXPLICIT: 3>

APPLICATION_ORDERED = <AcknowledgmentKind.APPLICATION_ORDERED:
2>

PROTOCOL = <AcknowledgmentKind.PROTOCOL: 0>

__eq__ (self: object, other: object) → bool

__getstate__ (self: object) → int

__hash__ (self: object) → int

__index__ (self: rti.connexdds.AcknowledgmentKind.AcknowledgmentKind) → int

__init__ (self: rti.connexdds.AcknowledgmentKind.AcknowledgmentKind, value: int) →
None

__int__ (self: rti.connexdds.AcknowledgmentKind.AcknowledgmentKind) → int

__members__ = {'APPLICATION_AUTO':
<AcknowledgmentKind.APPLICATION_AUTO: 1>,
'APPLICATION_EXPLICIT':
<AcknowledgmentKind.APPLICATION_EXPLICIT: 3>,
'APPLICATION_ORDERED': <AcknowledgmentKind.APPLICATION_ORDERED:
2>, 'PROTOCOL': <AcknowledgmentKind.PROTOCOL: 0>}

__module__ = 'rti.connexdds'

__ne__ (self: object, other: object) → bool

__repr__ (self: object) → str

__setstate__ (self: rti.connexdds.AcknowledgmentKind.AcknowledgmentKind, state: int)
→ None

__str__ ()
name(self: handle) -> str

property name
property value

PROTOCOL = <AcknowledgmentKind.PROTOCOL: 0>

__eq__ (self: rti.connexdds.AcknowledgmentKind, arg0: rti.connexdds.AcknowledgmentKind) →
bool

Apply operator to underlying enumerated values.

```

`__ge__` (*self*: rti.connextdds.AcknowledgmentKind, *arg0*: rti.connextdds.AcknowledgmentKind) → bool

Apply operator to underlying enumerated values.

`__gt__` (*self*: rti.connextdds.AcknowledgmentKind, *arg0*: rti.connextdds.AcknowledgmentKind) → bool

Apply operator to underlying enumerated values.

`__hash__` = None

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.AcknowledgmentKind) -> None`

Initializes enum to 0.

2. `__init__(self: rti.connextdds.AcknowledgmentKind, arg0: rti.connextdds.AcknowledgmentKind.AcknowledgmentKind) -> None`

Copy constructor.

`__int__` (*self*: rti.connextdds.AcknowledgmentKind) → *rti.connextdds.AcknowledgmentKind.AcknowledgmentKind*

Int conversion.

`__le__` (*self*: rti.connextdds.AcknowledgmentKind, *arg0*: rti.connextdds.AcknowledgmentKind) → bool

Apply operator to underlying enumerated values.

`__lt__` (*self*: rti.connextdds.AcknowledgmentKind, *arg0*: rti.connextdds.AcknowledgmentKind) → bool

Apply operator to underlying enumerated values.

`__module__` = `'rti.connextdds'`

`__ne__` (*self*: rti.connextdds.AcknowledgmentKind, *arg0*: rti.connextdds.AcknowledgmentKind) → bool

Apply operator to underlying enumerated values.

`__str__` (*self*: rti.connextdds.AcknowledgmentKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connextdds.**ActivityContextMask**

Bases: pybind11_object

ALL = 0000000000000000000000000000000000111111

DEFAULT = 0000000000000000000000000000000000111111

DOMAIN_ID = 000000000000000000000000000010000
ENTITY_KIND = 000000000000000000000000000001000
ENTITY_NAME = 000000000000000000000000000010000
GUID_PREFIX = 000000000000000000000000000000001
NONE = 000000000000000000000000000000000
TOPIC = 000000000000000000000000000000010
TYPE = 0000000000000000000000000000000100

__and__ (*self*: rti.connextdds.ActivityContextMask, *arg0*: rti.connextdds.ActivityContextMask) → *rti.connextdds.ActivityContextMask*

Bitwise logical AND of masks.

__bool__ (*self*: rti.connextdds.ActivityContextMask) → bool

Test if any bits are set.

__contains__ (*self*: rti.connextdds.ActivityContextMask, *arg0*: rti.connextdds.ActivityContextMask) → bool

__eq__ (*self*: rti.connextdds.ActivityContextMask, *arg0*: rti.connextdds.ActivityContextMask) → bool

Compare masks for equality.

__getitem__ (*self*: rti.connextdds.ActivityContextMask, *arg0*: int) → bool

Get individual mask bit.

__hash__ = None

__iand__ (*self*: rti.connextdds.ActivityContextMask, *arg0*: rti.connextdds.ActivityContextMask) → *rti.connextdds.ActivityContextMask*

Set mask to logical AND with another mask.

__ilshift__ (*self*: rti.connextdds.ActivityContextMask, *arg0*: int) → *rti.connextdds.ActivityContextMask*

Left shift bits in mask.

__init__ (**args*, ***kwargs*)

Overloaded function.

- __init__**(*self*: rti.connextdds.ActivityContextMask) -> None

Create an ActivityContextMask with no bits set.

- __init__**(*self*: rti.connextdds.ActivityContextMask, *value*: int) -> None

Creates a mask from the bits in an integer.

__int__ (*self*: rti.connexdds.ActivityContextMask) → int
Convert mask to int.

__ior__ (*self*: rti.connexdds.ActivityContextMask, *arg0*: rti.connexdds.ActivityContextMask) → *rti.connexdds.ActivityContextMask*
Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.ActivityContextMask, *arg0*: int) → *rti.connexdds.ActivityContextMask*
Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.ActivityContextMask, *arg0*: rti.connexdds.ActivityContextMask) → *rti.connexdds.ActivityContextMask*
Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.ActivityContextMask, *arg0*: int) → *rti.connexdds.ActivityContextMask*
Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ActivityContextMask, *arg0*: rti.connexdds.ActivityContextMask) → bool
Compare masks for inequality.

__or__ (*self*: rti.connexdds.ActivityContextMask, *arg0*: rti.connexdds.ActivityContextMask) → *rti.connexdds.ActivityContextMask*
Bitwise logical OR of masks.

__repr__ (*self*: rti.connexdds.ActivityContextMask) → str
Convert mask to string.

__rshift__ (*self*: rti.connexdds.ActivityContextMask, *arg0*: int) → *rti.connexdds.ActivityContextMask*
Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.ActivityContextMask, *arg0*: int, *arg1*: bool) → None
Set individual mask bit

__str__ (*self*: rti.connexdds.ActivityContextMask) → str
Convert mask to string.

__xor__ (*self*: rti.connexdds.ActivityContextMask, *arg0*: rti.connexdds.ActivityContextMask) → *rti.connexdds.ActivityContextMask*
Bitwise logical XOR of masks.

property count
Returns the number of bits set in the mask.

flip (*args, **kwargs)

Overloaded function.

1. flip(self: rti.connexdds.ActivityContextMask) -> rti.connexdds.ActivityContextMask

Flip all bits in the mask.

2. flip(self: rti.connexdds.ActivityContextMask, pos: int) -> rti.connexdds.ActivityContextMask

Flip the mask bit at the specified position.

reset (*args, **kwargs)

Overloaded function.

1. reset(self: rti.connexdds.ActivityContextMask) -> rti.connexdds.ActivityContextMask

Clear all bits in the mask.

2. reset(self: rti.connexdds.ActivityContextMask, pos: int) -> rti.connexdds.ActivityContextMask

Clear the mask bit at the specified position.

set (*args, **kwargs)

Overloaded function.

1. set(self: rti.connexdds.ActivityContextMask) -> rti.connexdds.ActivityContextMask

Set all bits in the mask.

2. set(self: rti.connexdds.ActivityContextMask, pos: int, value: bool = True) -> rti.connexdds.ActivityContextMask

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (self: rti.connexdds.ActivityContextMask, pos: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (self: rti.connexdds.ActivityContextMask) → bool

Test if all bits are set.

test_any (self: rti.connexdds.ActivityContextMask) → bool

Test if any bits are set.

test_none (self: rti.connexdds.ActivityContextMask) → bool

Test if none of the bits are set.

class rti.connexdds.**AliasType**

Bases: *DynamicType*

Represents a typedef.

__eq__ (*self*: rti.connexTdds.AliasType, *arg0*: rti.connexTdds.AliasType) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexTdds.AliasType, *name*: str, *type_name*: rti.connexTdds.DynamicType, *is_pointer*: bool = False) → None

Creates an alias with a name and a related type.

__module__ = 'rti.connexTdds'

__ne__ (*self*: rti.connexTdds.AliasType, *arg0*: rti.connexTdds.AliasType) → bool

Test for inequality.

property is_pointer

Gets whether this alias makes related_type a pointer

related_type (*self*: rti.connexTdds.AliasType) → rti.connexTdds.DynamicType

Gets the related type.

resolve (*self*: rti.connexTdds.DynamicType) → rti.connexTdds.DynamicType

Resolves an AliasType recursively to get the final underlying type.

static resolve_type (*alias_type*: rti.connexTdds.DynamicType) → rti.connexTdds.DynamicType

Resolves an AliasType recursively to get the final underlying type.

class rti.connexTdds.**AllocationSettings**

Bases: pybind11_object

AUTO_COUNT = -2

__eq__ (*self*: rti.connexTdds.AllocationSettings, *arg0*: rti.connexTdds.AllocationSettings) → bool

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexTdds.AllocationSettings) -> None

Creates an instance with an initial, max and incremental count set to zero.

2. **__init__**(*self*: rti.connexTdds.AllocationSettings, *initial_count*: int, *max_count*: int, *incremental_count*: int) -> None

Creates an instance with the given initial, maximum and incremental values.

__module__ = 'rti.connexTdds'

__ne__ (*self*: rti.connexTdds.AllocationSettings, *arg0*: rti.connexTdds.AllocationSettings) → bool

property incremental_count

Incremental count of resources.

property initial_count

Initial count of resources.

property max_count

Max count of resources.

exception rti.connextdds.AlreadyClosedError

Bases: *Exception*

__module__ = 'rti.connextdds'

class rti.connextdds.AnyDataReader

Bases: *IAnyDataReader*

__init__ (*self*: rti.connextdds.AnyDataReader, *reader*: rti.connextdds.IAnyDataReader) → None

Create an AnyDataReader instance from a object that conforms to the IAnyDataReader interface.

__module__ = 'rti.connextdds'

class rti.connextdds.AnyDataReaderListener

Bases: *pybind11_object*

__init__ (*self*: rti.connextdds.AnyDataReaderListener) → None

__module__ = 'rti.connextdds'

on_data_available (*self*: rti.connextdds.AnyDataReaderListener, *arg0*:
rti.connextdds.AnyDataReader) → None

Data available callback.

on_liveliness_changed (*self*: rti.connextdds.AnyDataReaderListener, *arg0*:
rti.connextdds.AnyDataReader, *arg1*:
rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connextdds.AnyDataReaderListener, *arg0*:
rti.connextdds.AnyDataReader, *arg1*:
rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connextdds.AnyDataReaderListener, *arg0*:
rti.connextdds.AnyDataReader, *arg1*:
rti.connextdds.RequestedIncompatibleQosStatus) →
None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connextdds.AnyDataReaderListener, *arg0*:
rti.connextdds.AnyDataReader, *arg1*: rti.connextdds.SampleLostStatus) →
None

Sample lost callback.

on_sample_rejected (*self*: rti.connextdds.AnyDataReaderListener, *arg0*: rti.connextdds.AnyDataReader, *arg1*: rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connextdds.AnyDataReaderListener, *arg0*: rti.connextdds.AnyDataReader, *arg1*: rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class rti.connextdds.**AnyDataReaderSeq**

Bases: pybind11_object

__add__ (*self*: rti.connextdds.AnyDataReaderSeq, *arg0*: rti.connextdds.AnyDataReaderSeq) → *rti.connextdds.AnyDataReaderSeq*

__bool__ (*self*: rti.connextdds.AnyDataReaderSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connextdds.AnyDataReaderSeq, *x*: rti.connextdds.AnyDataReader) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connextdds.AnyDataReaderSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connextdds.AnyDataReaderSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connextdds.AnyDataReaderSeq, *arg0*: rti.connextdds.AnyDataReaderSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connextdds.AnyDataReaderSeq, *s*: slice) -> rti.connextdds.AnyDataReaderSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connextdds.AnyDataReaderSeq, *arg0*: int) -> rti.connextdds.AnyDataReader

__hash__ = None

__iadd__ (*self*: rti.connextdds.AnyDataReaderSeq, *arg0*: rti.connextdds.AnyDataReaderSeq) → *rti.connextdds.AnyDataReaderSeq*

__imul__ (*self*: rti.connextdds.AnyDataReaderSeq, *arg0*: int) → *rti.connextdds.AnyDataReaderSeq*

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.AnyDataReaderSeq) -> None
2. **__init__**(self: rti.connexdds.AnyDataReaderSeq, arg0: rti.connexdds.AnyDataReaderSeq) -> None

Copy constructor

3. **__init__**(self: rti.connexdds.AnyDataReaderSeq, arg0: Iterable) -> None

__iter__ (self: rti.connexdds.AnyDataReaderSeq) → Iterator

__len__ (self: rti.connexdds.AnyDataReaderSeq) → int

__module__ = 'rti.connexdds'

__mul__ (self: rti.connexdds.AnyDataReaderSeq, arg0: int) → rti.connexdds.AnyDataReaderSeq

__ne__ (self: rti.connexdds.AnyDataReaderSeq, arg0: rti.connexdds.AnyDataReaderSeq) → bool

__rmul__ (self: rti.connexdds.AnyDataReaderSeq, arg0: int) → rti.connexdds.AnyDataReaderSeq

__setitem__ (*args, **kwargs)

Overloaded function.

1. **__setitem__**(self: rti.connexdds.AnyDataReaderSeq, arg0: int, arg1: rti.connexdds.AnyDataReader) -> None
2. **__setitem__**(self: rti.connexdds.AnyDataReaderSeq, arg0: slice, arg1: rti.connexdds.AnyDataReaderSeq) -> None

Assign list elements using a slice object

append (self: rti.connexdds.AnyDataReaderSeq, x: rti.connexdds.AnyDataReader) → None

Add an item to the end of the list

clear (self: rti.connexdds.AnyDataReaderSeq) → None

Clear the contents

count (self: rti.connexdds.AnyDataReaderSeq, x: rti.connexdds.AnyDataReader) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. **extend**(self: rti.connexdds.AnyDataReaderSeq, L: rti.connexdds.AnyDataReaderSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(self: rti.connexdds.AnyDataReaderSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.AnyDataReaderSeq, *i*: int, *x*: rti.connexdds.AnyDataReader) → None
 Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.AnyDataReaderSeq) -> rti.connexdds.AnyDataReader

Remove and return the last item

2. pop(*self*: rti.connexdds.AnyDataReaderSeq, *i*: int) -> rti.connexdds.AnyDataReader

Remove and return the item at index *i*

remove (*self*: rti.connexdds.AnyDataReaderSeq, *x*: rti.connexdds.AnyDataReader) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.AnyDataWriter

Bases: *IAnyDataWriter*

__init__ (*self*: rti.connexdds.AnyDataWriter, *writer*: rti.connexdds.IAnyDataWriter) → None

Create an AnyDataWriter instance from a object that conforms to the IAnyDataWriter interface.

__module__ = 'rti.connexdds'

class rti.connexdds.AnyDataWriterListener

Bases: pybind11_object

__init__ (*self*: rti.connexdds.AnyDataWriterListener) → None

__module__ = 'rti.connexdds'

on_application_acknowledgment (*self*: rti.connexdds.AnyDataWriterListener, *arg0*:
 rti.connexdds.AnyDataWriter, *arg1*:
 rti.connexdds.AcknowledgmentInfo) → None

On application acknowledgment callback

on_instance_replaced (*self*: rti.connexdds.AnyDataWriterListener, *arg0*:
 rti.connexdds.AnyDataWriter, *arg1*: rti.connexdds.InstanceHandle)
 → None

On instance replaced callback.

on_liveliness_lost (*self*: rti.connexdds.AnyDataWriterListener, *arg0*:
 rti.connexdds.AnyDataWriter, *arg1*:
 rti.connexdds.LivelinessLostStatus) → None

Liveliness lost callback.

on_offered_deadline_missed (*self*: rti.connexdds.AnyDataWriterListener, *arg0*:
 rti.connexdds.AnyDataWriter, *arg1*:
 rti.connexdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

on_offered_incompatible_qos (*self*: rti.connextdds.AnyDataWriterListener, *arg0*: rti.connextdds.AnyDataWriter, *arg1*: rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

on_publication_matched (*self*: rti.connextdds.AnyDataWriterListener, *arg0*: rti.connextdds.AnyDataWriter, *arg1*: rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connextdds.AnyDataWriterListener, *arg0*: rti.connextdds.AnyDataWriter, *arg1*: rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connextdds.AnyDataWriterListener, *arg0*: rti.connextdds.AnyDataWriter, *arg1*: rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connextdds.AnyDataWriterListener, *arg0*: rti.connextdds.AnyDataWriter, *arg1*: rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class rti.connextdds.AnyDataWriterSeq

Bases: pybind11_object

__add__ (*self*: rti.connextdds.AnyDataWriterSeq, *arg0*: rti.connextdds.AnyDataWriterSeq) → rti.connextdds.AnyDataWriterSeq

__bool__ (*self*: rti.connextdds.AnyDataWriterSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connextdds.AnyDataWriterSeq, *x*: rti.connextdds.AnyDataWriter) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connextdds.AnyDataWriterSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connextdds.AnyDataWriterSeq, *arg0*: slice) -> None

Delete list elements using a slice object

`__eq__` (*self*: rti.connexdds.AnyDataWriterSeq, *arg0*: rti.connexdds.AnyDataWriterSeq) → bool

`__getitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__(self: rti.connexdds.AnyDataWriterSeq, s: slice) -> rti.connexdds.AnyDataWriterSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.AnyDataWriterSeq, arg0: int) -> rti.connexdds.AnyDataWriter`

`__hash__` = None

`__iadd__` (*self*: rti.connexdds.AnyDataWriterSeq, *arg0*: rti.connexdds.AnyDataWriterSeq) → *rti.connexdds.AnyDataWriterSeq*

`__imul__` (*self*: rti.connexdds.AnyDataWriterSeq, *arg0*: int) → *rti.connexdds.AnyDataWriterSeq*

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.AnyDataWriterSeq) -> None`
2. `__init__(self: rti.connexdds.AnyDataWriterSeq, arg0: rti.connexdds.AnyDataWriterSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.AnyDataWriterSeq, arg0: Iterable) -> None`

`__iter__` (*self*: rti.connexdds.AnyDataWriterSeq) → Iterator

`__len__` (*self*: rti.connexdds.AnyDataWriterSeq) → int

`__module__` = 'rti.connexdds'

`__mul__` (*self*: rti.connexdds.AnyDataWriterSeq, *arg0*: int) → *rti.connexdds.AnyDataWriterSeq*

`__ne__` (*self*: rti.connexdds.AnyDataWriterSeq, *arg0*: rti.connexdds.AnyDataWriterSeq) → bool

`__rmul__` (*self*: rti.connexdds.AnyDataWriterSeq, *arg0*: int) → *rti.connexdds.AnyDataWriterSeq*

`__setitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__setitem__(self: rti.connexdds.AnyDataWriterSeq, arg0: int, arg1: rti.connexdds.AnyDataWriter) -> None`
2. `__setitem__(self: rti.connexdds.AnyDataWriterSeq, arg0: slice, arg1: rti.connexdds.AnyDataWriterSeq) -> None`

Assign list elements using a slice object

append (*self*: rti.connexdds.AnyDataWriterSeq, *x*: rti.connexdds.AnyDataWriter) → None
 Add an item to the end of the list

clear (*self*: rti.connexdds.AnyDataWriterSeq) → None
 Clear the contents

count (*self*: rti.connexdds.AnyDataWriterSeq, *x*: rti.connexdds.AnyDataWriter) → int
 Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)
 Overloaded function.

1. extend(*self*: rti.connexdds.AnyDataWriterSeq, *L*: rti.connexdds.AnyDataWriterSeq) -> None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connexdds.AnyDataWriterSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.AnyDataWriterSeq, *i*: int, *x*: rti.connexdds.AnyDataWriter) → None
 Insert an item at a given position.

pop (**args*, ***kwargs*)
 Overloaded function.

1. pop(*self*: rti.connexdds.AnyDataWriterSeq) -> rti.connexdds.AnyDataWriter

Remove and return the last item

2. pop(*self*: rti.connexdds.AnyDataWriterSeq, *i*: int) -> rti.connexdds.AnyDataWriter

Remove and return the item at index *i*

remove (*self*: rti.connexdds.AnyDataWriterSeq, *x*: rti.connexdds.AnyDataWriter) → None
 Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.**AnyTopic**

Bases: *IAnyTopic*

__init__ (*self*: rti.connexdds.AnyTopic, *topic*: rti.connexdds.IAnyTopic) → None
 Create an AnyTopic instance from a object that conforms to the IAnyTopic interface.

__module__ = 'rti.connexdds'

class rti.connexdds.**AnyTopicListener**

Bases: *pybind11_object*

__init__ (*self*: rti.connexdds.AnyTopicListener) → None

__module__ = 'rti.connexdds'

on_inconsistent_topic (*self*: rti.connexdds.AnyTopicListener, *arg0*: rti.connexdds.AnyTopic, *arg1*: rti.connexdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class rti.connexdds.**AnyTopicSeq**

Bases: pybind11_object

__add__ (*self*: rti.connexdds.AnyTopicSeq, *arg0*: rti.connexdds.AnyTopicSeq) → *rti.connexdds.AnyTopicSeq*

__bool__ (*self*: rti.connexdds.AnyTopicSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.AnyTopicSeq, *x*: rti.connexdds.AnyTopic) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.AnyTopicSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.AnyTopicSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.AnyTopicSeq, *arg0*: rti.connexdds.AnyTopicSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.AnyTopicSeq, *s*: slice) -> rti.connexdds.AnyTopicSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.AnyTopicSeq, *arg0*: int) -> rti.connexdds.AnyTopic

__hash__ = None

__iadd__ (*self*: rti.connexdds.AnyTopicSeq, *arg0*: rti.connexdds.AnyTopicSeq) → *rti.connexdds.AnyTopicSeq*

__imul__ (*self*: rti.connexdds.AnyTopicSeq, *arg0*: int) → *rti.connexdds.AnyTopicSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.AnyTopicSeq) -> None

2. **__init__**(*self*: rti.connexdds.AnyTopicSeq, *arg0*: rti.connexdds.AnyTopicSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.AnyTopicSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connextdds.AnyTopicSeq) → Iterator

__len__ (*self*: rti.connextdds.AnyTopicSeq) → int

__module__ = 'rti.connextdds'

__mul__ (*self*: rti.connextdds.AnyTopicSeq, *arg0*: int) → rti.connextdds.AnyTopicSeq

__ne__ (*self*: rti.connextdds.AnyTopicSeq, *arg0*: rti.connextdds.AnyTopicSeq) → bool

__rmul__ (*self*: rti.connextdds.AnyTopicSeq, *arg0*: int) → rti.connextdds.AnyTopicSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connextdds.AnyTopicSeq, *arg0*: int, *arg1*: rti.connextdds.AnyTopic) -> None
2. **__setitem__**(*self*: rti.connextdds.AnyTopicSeq, *arg0*: slice, *arg1*: rti.connextdds.AnyTopicSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connextdds.AnyTopicSeq, *x*: rti.connextdds.AnyTopic) → None

Add an item to the end of the list

clear (*self*: rti.connextdds.AnyTopicSeq) → None

Clear the contents

count (*self*: rti.connextdds.AnyTopicSeq, *x*: rti.connextdds.AnyTopic) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connextdds.AnyTopicSeq, *L*: rti.connextdds.AnyTopicSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connextdds.AnyTopicSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.AnyTopicSeq, *i*: int, *x*: rti.connextdds.AnyTopic) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connextdds.AnyTopicSeq) -> rti.connextdds.AnyTopic

Remove and return the last item

2. **pop**(*self*: rti.connextdds.AnyTopicSeq, *i*: int) -> rti.connextdds.AnyTopic

Remove and return the item at index *i*

remove (*self*: rti.connexdds.AnyTopicSeq, *x*: rti.connexdds.AnyTopic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.**ArrayType**

Bases: *CollectionType*

__eq__ (*self*: rti.connexdds.ArrayType, *arg0*: rti.connexdds.ArrayType) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ArrayType, *data_type*: rti.connexdds.DynamicType, *dimension*: int) -> None

Creates a unidimensional array.

2. **__init__**(*self*: rti.connexdds.ArrayType, *data_type*: rti.connexdds.DynamicType, *dimensions*: rti.connexdds.Int32Seq) -> None

Create a multidimensional array.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ArrayType, *arg0*: rti.connexdds.ArrayType) → bool

Test for inequality.

dimension (*self*: rti.connexdds.ArrayType, *index*: int) → int

Returns the size of the *i*th dimension

property dimension_count

Number of dimensions.

property total_element_count

Total element count across all dimensions.

class rti.connexdds.**AsynchronousPublisher**

Bases: *pybind11_object*

__eq__ (*self*: rti.connexdds.AsynchronousPublisher, *arg0*: rti.connexdds.AsynchronousPublisher) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.AsynchronousPublisher) → None

Create a default AsynchronousPublisher QoS policy.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connextdds.AsynchronousPublisher, *arg0*: rti.connextdds.AsynchronousPublisher) → bool

Test for inequality.

property asynchronous_batch_thread

Get/set the settings associated with the asynchronous batch flush thread.

property disable_asynchronous_batch

Get/set the asynchronous batch flushing capability for this policy.

property disable_asynchronous_write

Get/set the asynchronous write capability for this policy.

property disable_topic_query_publication

Get/set the topic query publication capability for this policy.

disabled = <rti.connextdds.AsynchronousPublisher object>

static enabled (*enable_async_batch*: bool = False) → rti.connextdds.AsynchronousPublisher

Create an AsynchronousPublisher QoS policy with asynchronous write (and optionally asynchronous batch flushing) enabled.

property thread

Get/set the settings associated with the asynchronous publisher thread.

property topic_query_publication_thread

Get/set the settings associated with the topic query publication thread.

class rti.connextdds.Availability

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.Availability, *arg0*: rti.connextdds.Availability) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.Availability) -> None

Create a default Availability QoS policy.

2. **__init__**(*self*: rti.connextdds.Availability, *enable_required_subscriptions*: bool, *data_waiting_time*: rti.connextdds.Duration, *endpoint_waiting_time*: rti.connextdds.Duration, *required_endpoint_groups*: rti.connextdds.EndpointGroupVector) -> None

Create an Availability QoS policy with the provided parameters.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.Availability, *arg0*: rti.connextdds.Availability) → bool

Test for inequality.

property enable_required_subscriptions

Get/set the status of required subscriptions for this policy.

property max_data_availability_waiting_time

Get/set the amount of time to wait before delivering a sample to the application without having received some of the previous samples.

property max_endpoint_availability_waiting_time

Get/set the amount of time to wait to discover DataWriters providing samples for the same data source (virtual GUID).

property required_matched_endpoint_groups

Get/set a copy of the required endpoint groups.

class rti.connextdds.Batch

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.Batch, *arg0*: rti.connextdds.Batch) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.Batch) → None

Create a Batch QoS policy with the default settings (disabled).

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.Batch, *arg0*: rti.connextdds.Batch) → bool

Test for inequality.

disabled = <rti.connextdds.Batch object>

property enable

Get/set boolean that specifies whether batching is enabled.

enabled = <rti.connextdds.Batch object>

static enabled_with_max_data_bytes (*max_data_bytes*: int) → rti.connextdds.Batch

Create a Batch policy with batching enabled that will flush based on the cumulative size of serialized samples.

static enabled_with_max_samples (*max_samples*: int) → rti.connextdds.Batch

Create a Batch policy with batching enabled that will flush based on the number of samples written to the batch.

property max_data_bytes

Get/set the maximum cumulative size of serialized samples in a batch. When this limit is reached the batch will be flushed.

property max_flush_delay

Get/set the maximum delay to flush a batch, measured from the time the first sample in the batch is written.

property max_samples

Get/set the maximum number of serialized samples in a batch. When this limit is reached the batch will be flushed.

property source_timestamp_resolution

Get/set the timestamp resolution of the samples in the batch.

property thread_safe_write

Get/set boolean that specifies whether batch writes must be thread safe.

class `rti.connextdds.BoolSeq`

Bases: `pybind11_object`

__add__ (*self*: `rti.connextdds.BoolSeq`, *arg0*: `rti.connextdds.BoolSeq`) → *rti.connextdds.BoolSeq*

__bool__ (*self*: `rti.connextdds.BoolSeq`) → `bool`

Check whether the list is nonempty

__contains__ (*self*: `rti.connextdds.BoolSeq`, *x*: *bool*) → `bool`

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: `rti.connextdds.BoolSeq`, *arg0*: `int`) -> `None`

Delete the list elements at index *i*

2. **__delitem__**(*self*: `rti.connextdds.BoolSeq`, *arg0*: `slice`) -> `None`

Delete list elements using a slice object

__eq__ (*self*: `rti.connextdds.BoolSeq`, *arg0*: `rti.connextdds.BoolSeq`) → `bool`

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: `rti.connextdds.BoolSeq`, *s*: `slice`) -> `rti.connextdds.BoolSeq`

Retrieve list elements using a slice object

2. **__getitem__**(*self*: `rti.connextdds.BoolSeq`, *arg0*: `int`) -> `bool`

__hash__ = `None`

__iadd__ (*self*: `rti.connextdds.BoolSeq`, *arg0*: `rti.connextdds.BoolSeq`) → *rti.connextdds.BoolSeq*

__imul__ (*self*: `rti.connextdds.BoolSeq`, *arg0*: `int`) → *rti.connextdds.BoolSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: `rti.connextdds.BoolSeq`) -> `None`

2. **__init__**(*self*: `rti.connextdds.BoolSeq`, *arg0*: `rti.connextdds.BoolSeq`) -> `None`

Copy constructor

3. `__init__(self: rti.connextdds.BoolSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.BoolSeq) → Iterator`

`__len__(self: rti.connextdds.BoolSeq) → int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.BoolSeq, arg0: int) → rti.connextdds.BoolSeq`

`__ne__(self: rti.connextdds.BoolSeq, arg0: rti.connextdds.BoolSeq) → bool`

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>`

`__repr__(self: rti.connextdds.BoolSeq) → str`

Return the canonical string representation of this list.

`__rmul__(self: rti.connextdds.BoolSeq, arg0: int) → rti.connextdds.BoolSeq`

`__setitem__(*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connextdds.BoolSeq, arg0: int, arg1: bool) -> None`

2. `__setitem__(self: rti.connextdds.BoolSeq, arg0: slice, arg1: rti.connextdds.BoolSeq) -> None`

Assign list elements using a slice object

`append(self: rti.connextdds.BoolSeq, x: bool) → None`

Add an item to the end of the list

`clear(self: rti.connextdds.BoolSeq) → None`

Clear the contents

`count(self: rti.connextdds.BoolSeq, x: bool) → int`

Return the number of times `x` appears in the list

`extend(*args, **kwargs)`

Overloaded function.

1. `extend(self: rti.connextdds.BoolSeq, L: rti.connextdds.BoolSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.BoolSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

`insert(self: rti.connextdds.BoolSeq, i: int, x: bool) → None`

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connextdds.BoolSeq) -> bool

Remove and return the last item

2. pop(self: rti.connextdds.BoolSeq, i: int) -> bool

Remove and return the item at index i

remove (self: rti.connextdds.BoolSeq, x: bool) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class rti.connextdds.BoolType

Bases: *DynamicType*

__eq__ (self: rti.connextdds.BoolType, arg0: rti.connextdds.BoolType) → bool

Test for equality.

__hash__ = None

__init__ (self: rti.connextdds.BoolType) → None

Get the singleton for BoolType

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.BoolType, arg0: rti.connextdds.BoolType) → bool

Test for inequality.

class rti.connextdds.BuiltinProfiles

Bases: pybind11_object

__init__ (*args, **kwargs)

__module__ = 'rti.connextdds'

baseline = 'BuiltinQosLib::Baseline'

baseline_5_0_0 = 'BuiltinQosLib::Baseline.5.0.0'

baseline_5_1_0 = 'BuiltinQosLib::Baseline.5.1.0'

baseline_5_2_0 = 'BuiltinQosLib::Baseline.5.2.0'

baseline_5_3_0 = 'BuiltinQosLib::Baseline.5.3.0'

baseline_6_0_0 = 'BuiltinQosLib::Baseline.6.0.0'

generic_auto_tuning = 'BuiltinQosLib::Generic.AutoTuning'

generic_best_effort = 'BuiltinQosLib::Generic.BestEffort'

generic_common = 'BuiltinQosLib::Generic.Common'

```
generic_connext_micro_compatibility =
'BuiltinQosLib::Generic.ConnexMicroCompatibility'

generic_connext_micro_compatibility_2_4_3 =
'BuiltinQosLib::Generic.ConnexMicroCompatibility.2.4.3'

generic_connext_micro_compatibility_2_4_9 =
'BuiltinQosLib::Generic.ConnexMicroCompatibility.2.4.9'

generic_keep_last_reliable =
'BuiltinQosLib::Generic.KeepLastReliable'

generic_keep_last_reliable_large_data =
'BuiltinQosLib::Generic.KeepLastReliable.LargeData'

generic_keep_last_reliable_large_data_fast_flow =
'BuiltinQosLib::Generic.KeepLastReliable.LargeData.FastFlow'

generic_keep_last_reliable_large_data_medium_flow =
'BuiltinQosLib::Generic.KeepLastReliable.LargeData.MediumFlow'

generic_keep_last_reliable_large_data_slow_flow =
'BuiltinQosLib::Generic.KeepLastReliable.LargeData.SlowFlow'

generic_keep_last_reliable_persistent =
'BuiltinQosLib::Generic.KeepLastReliable.Persistent'

generic_keep_last_reliable_transient =
'BuiltinQosLib::Generic.KeepLastReliable.Transient'

generic_keep_last_reliable_transient_local =
'BuiltinQosLib::Generic.KeepLastReliable.TransientLocal'

generic_minimal_memory_footprint =
'BuiltinQosLib::Generic.MinimalMemoryFootprint'

generic_monitoring_common =
'BuiltinQosLib::Generic.Monitoring.Common'

generic_other_dds_vendor_compatibility =
'BuiltinQosLib::Generic.510TransportCompatibility'

generic_participant_large_data =
'BuiltinQosLib::Generic.Participant.LargeData'

generic_participant_large_data_monitoring =
'BuiltinQosLib::Generic.Participant.LargeData.Monitoring'

generic_security = 'BuiltinQosLib::Generic.Security'

generic_strict_reliable = 'BuiltinQosLib::Generic.StrictReliable'
```

```

generic_strict_reliable_high_throughput =
    'BuiltinQosLib::Generic.StrictReliable.HighThroughput'

generic_strict_reliable_large_data =
    'BuiltinQosLib::Generic.StrictReliable.LargeData'

generic_strict_reliable_large_data_fast_flow =
    'BuiltinQosLib::Generic.StrictReliable.LargeData.FastFlow'

generic_strict_reliable_large_data_medium_flow =
    'BuiltinQosLib::Generic.StrictReliable.LargeData.MediumFlow'

generic_strict_reliable_large_data_slow_flow =
    'BuiltinQosLib::Generic.StrictReliable.LargeData.SlowFlow'

generic_strict_reliable_low_latency =
    'BuiltinQosLib::Generic.StrictReliable.LowLatency'

library_name = 'BuiltinQosLib'

pattern_alarm_event = 'BuiltinQosLib::Pattern.AlarmEvent'
pattern_alarm_status = 'BuiltinQosLib::Pattern.AlarmStatus'
pattern_event = 'BuiltinQosLib::Pattern.Event'
pattern_last_value_cache = 'BuiltinQosLib::Pattern.LastValueCache'
pattern_periodic_data = 'BuiltinQosLib::Pattern.PeriodicData'
pattern_reliable_streaming =
    'BuiltinQosLib::Pattern.ReliableStreaming'
pattern_status = 'BuiltinQosLib::Pattern.Status'
pattern_streaming = 'BuiltinQosLib::Pattern.Streaming'

```

```
class rti.connextdds.BuiltinTopicKey
```

```
    Bases: pybind11_object
```

```
    __eq__(self: rti.connextdds.BuiltinTopicKey, arg0: rti.connextdds.BuiltinTopicKey) → bool
        Test for equality.
```

```
    __hash__ = None
```

```
    __init__(self: rti.connextdds.BuiltinTopicKey) → None
        Creates a key whose value is all zeros.
```

```
    __module__ = 'rti.connextdds'
```

```
    __ne__(self: rti.connextdds.BuiltinTopicKey, arg0: rti.connextdds.BuiltinTopicKey) → bool
        Test for inequality.
```

`__repr__` (*self*: rti.connexdds.BuiltinTopicKey) → str

Returns a string representation of the key.

property value

Returns a copy of the four integers that represent the key.

class rti.connexdds.BuiltinTopicReaderResourceLimits

Bases: pybind11_object

`__eq__` (*self*: rti.connexdds.BuiltinTopicReaderResourceLimits, *arg0*: rti.connexdds.BuiltinTopicReaderResourceLimits) → bool

`__hash__` = None

`__init__` (*self*: rti.connexdds.BuiltinTopicReaderResourceLimits) → None

Create a policy with default settings.

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.BuiltinTopicReaderResourceLimits, *arg0*: rti.connexdds.BuiltinTopicReaderResourceLimits) → bool

property disable_fragmentation_support

Determines whether the DataReader can receive fragmented samples.

property dynamically_allocate_fragmented_samples

Determines whether the DataReader pre-allocates storage for storing fragmented samples.

property initial_fragmented_samples

The initial number of samples for which a DataReader may store fragments.

property initial_infos

Initial number of sample infos.

property initial_outstanding_reads

The initial number of outstanding calls to read/take (or one of their variants) on the same DataReader for which memory has not been returned by calling `LoanedSamples.return_loan()`.

property initial_samples

Initial number of samples

property max_fragmented_samples

The maximum number of samples for which the DataReader may store fragments at a given point in time.

property max_fragmented_samples_per_remote_writer

The maximum number of samples per remote writer for which a built-in topic reader may store fragments.

property max_fragments_per_sample

Maximum number of fragments for a single sample.

property max_infos

Maximum number of sample infos.

property max_outstanding_reads

The max number of outstanding calls to read/take (or one of their variants) on the same DataReader for which memory has not been returned by calling LoanedSamples.return_loan().

property max_samples

Maximum number of samples

property max_samples_per_read

The maximum number of data samples that the application can receive from the middleware in a single call to DataReader.read() or DataReader.take(). If more data exists in the middleware, the application will need to issue multiple read/take calls.

class rti.connexdds.ByteVector

Bases: pybind11_object

A DDS standard container with functionality similar to a C++ vector.

__eq__ (*self*: rti.connexdds.ByteVector, *arg0*: rti.connexdds.ByteVector) → bool

Compare ByteVectors for equality.

__getitem__ (*self*: rti.connexdds.ByteVector, *arg0*: int) → int

Get the value at the specified index.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ByteVector, *buffer*: buffer) -> None

Create a ByteVector from another Python buffer.

2. **__init__**(*self*: rti.connexdds.ByteVector) -> None

Create an empty ByteVector

3. **__init__**(*self*: rti.connexdds.ByteVector, *size*: int) -> None

Create a ByteVector with a preallocated size.

4. **__init__**(*self*: rti.connexdds.ByteVector, *vector*: rti.connexdds.ByteVector) -> None

Create a copy from another ByteVector.

5. **__init__**(*self*: rti.connexdds.ByteVector, *arg0*: Iterable) -> None

6. **__init__**(*self*: rti.connexdds.ByteVector, *list*: rti.connexdds.Uint8Seq) -> None

Create a ByteVector from a list of values.

__iter__ (*self*: rti.connexdds.ByteVector) → Iterator

Iterate over the contents of the vector.

`__len__` (*self*: rti.connexdds.ByteVector) → int

Get the length of the ByteVector.

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.ByteVector, *arg0*: rti.connexdds.ByteVector) → bool

Compare ByteVectors for inequality.

`__setitem__` (*self*: rti.connexdds.ByteVector, *arg0*: int, *arg1*: int) → None

Set the value at the specified index.

`clear` (*self*: rti.connexdds.ByteVector) → None

Resize ByteVector to 0.

`resize` (*self*: rti.connexdds.ByteVector, *size*: int) → None

Resize ByteVector.

class rti.connexdds.CdrPaddingKind

Bases: pybind11_object

AUTO = <CdrPaddingKind.AUTO: 2>

class CdrPaddingKind

Bases: pybind11_object

Members:

ZERO : Set padding bytes to zero during CDR serialization.

NOT_SET : Don't set padding bytes to any value.

AUTO : Set the value automatically, depending on the Entity.

When set on a DomainParticipant the default behavior is NOT_SET

When set on a DataWriter or DataReader the behavior is to inherit the value from the DomainParticipant.

AUTO = <CdrPaddingKind.AUTO: 2>

NOT_SET = <CdrPaddingKind.NOT_SET: 1>

ZERO = <CdrPaddingKind.ZERO: 0>

`__eq__` (*self*: object, *other*: object) → bool

`__getstate__` (*self*: object) → int

`__hash__` (*self*: object) → int

`__index__` (*self*: rti.connexdds.CdrPaddingKind.CdrPaddingKind) → int

`__init__` (*self*: rti.connexdds.CdrPaddingKind.CdrPaddingKind, *value*: int) → None

`__int__` (*self*: rti.connexdds.CdrPaddingKind.CdrPaddingKind) → int


```
__members__ = {'AUTO': <CdrPaddingKind.AUTO: 2>, 'NOT_SET':
<CdrPaddingKind.NOT_SET: 1>, 'ZERO': <CdrPaddingKind.ZERO: 0>}
```

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: object, other: object) → bool
```

```
__repr__ (self: object) → str
```

```
__setstate__ (self: rti.connexdds.CdrPaddingKind.CdrPaddingKind, state: int) → None
```

```
__str__ ()
```

```
name(self: handle) -> str
```

property name

property value

```
NOT_SET = <CdrPaddingKind.NOT_SET: 1>
```

```
ZERO = <CdrPaddingKind.ZERO: 0>
```

```
__eq__ (self: rti.connexdds.CdrPaddingKind, arg0: rti.connexdds.CdrPaddingKind) → bool
```

Apply operator to underlying enumerated values.

```
__ge__ (self: rti.connexdds.CdrPaddingKind, arg0: rti.connexdds.CdrPaddingKind) → bool
```

Apply operator to underlying enumerated values.

```
__gt__ (self: rti.connexdds.CdrPaddingKind, arg0: rti.connexdds.CdrPaddingKind) → bool
```

Apply operator to underlying enumerated values.

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connexdds.CdrPaddingKind) -> None`

Initializes enum to 0.

2. `__init__(self: rti.connexdds.CdrPaddingKind, arg0: rti.connexdds.CdrPaddingKind.CdrPaddingKind) -> None`

Copy constructor.

```
__int__ (self: rti.connexdds.CdrPaddingKind) → rti.connexdds.CdrPaddingKind.CdrPaddingKind
```

Int conversion.

```
__le__ (self: rti.connexdds.CdrPaddingKind, arg0: rti.connexdds.CdrPaddingKind) → bool
```

Apply operator to underlying enumerated values.

```
__lt__ (self: rti.connexdds.CdrPaddingKind, arg0: rti.connexdds.CdrPaddingKind) → bool
```

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.CdrPaddingKind, *arg0*: rti.connexdds.CdrPaddingKind) → bool
Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.CdrPaddingKind) → str
String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.ChannelSettings

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.ChannelSettings, *arg0*: rti.connexdds.ChannelSettings) → bool
Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.ChannelSettings, *multicast_settings*:
rti.connexdds.TransportMulticastSettingsSeq, *filter_expression*: str, *priority*: int) →
None

Creates an instance with the specified multicast settings, filter expression and priority.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ChannelSettings, *arg0*: rti.connexdds.ChannelSettings) → bool
Test for inequality.

property filter_expression

A logical expression used to determine the data that will be published in the channel.

property multicast_settings

A sequence of TransportMulticastSettings used to configure the multicast addresses associated with a channel.

property priority

Publication priority.

class rti.connexdds.ChannelSettingsSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.ChannelSettingsSeq, *arg0*: rti.connexdds.ChannelSettingsSeq) →
rti.connexdds.ChannelSettingsSeq

__bool__ (*self*: rti.connexdds.ChannelSettingsSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.ChannelSettingsSeq, *x*: rti.connexdds.ChannelSettings) →
bool

Return true the container contains *x*

`__delitem__` (*args, **kwargs)

Overloaded function.

1. `__delitem__(self: rti.connextdds.ChannelSettingsSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connextdds.ChannelSettingsSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (self: rti.connextdds.ChannelSettingsSeq, arg0: rti.connextdds.ChannelSettingsSeq) → bool

`__getitem__` (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connextdds.ChannelSettingsSeq, s: slice) -> rti.connextdds.ChannelSettingsSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.ChannelSettingsSeq, arg0: int) -> rti.connextdds.ChannelSettings`

`__hash__` = None

`__iadd__` (self: rti.connextdds.ChannelSettingsSeq, arg0: rti.connextdds.ChannelSettingsSeq) → *rti.connextdds.ChannelSettingsSeq*

`__imul__` (self: rti.connextdds.ChannelSettingsSeq, arg0: int) → *rti.connextdds.ChannelSettingsSeq*

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.ChannelSettingsSeq) -> None`
2. `__init__(self: rti.connextdds.ChannelSettingsSeq, arg0: rti.connextdds.ChannelSettingsSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.ChannelSettingsSeq, arg0: Iterable) -> None`

`__iter__` (self: rti.connextdds.ChannelSettingsSeq) → Iterator

`__len__` (self: rti.connextdds.ChannelSettingsSeq) → int

`__module__` = 'rti.connextdds'

`__mul__` (self: rti.connextdds.ChannelSettingsSeq, arg0: int) → *rti.connextdds.ChannelSettingsSeq*

`__ne__` (self: rti.connextdds.ChannelSettingsSeq, arg0: rti.connextdds.ChannelSettingsSeq) → bool

`__rmul__` (self: rti.connextdds.ChannelSettingsSeq, arg0: int) → *rti.connextdds.ChannelSettingsSeq*

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.ChannelSettingsSeq, arg0: int, arg1: rti.connextdds.ChannelSettings) -> None`
2. `__setitem__(self: rti.connextdds.ChannelSettingsSeq, arg0: slice, arg1: rti.connextdds.ChannelSettingsSeq) -> None`

Assign list elements using a slice object

append (self: rti.connextdds.ChannelSettingsSeq, x: rti.connextdds.ChannelSettings) → None

Add an item to the end of the list

clear (self: rti.connextdds.ChannelSettingsSeq) → None

Clear the contents

count (self: rti.connextdds.ChannelSettingsSeq, x: rti.connextdds.ChannelSettings) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connextdds.ChannelSettingsSeq, L: rti.connextdds.ChannelSettingsSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.ChannelSettingsSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.ChannelSettingsSeq, i: int, x: rti.connextdds.ChannelSettings) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.ChannelSettingsSeq) -> rti.connextdds.ChannelSettings`

Remove and return the last item

2. `pop(self: rti.connextdds.ChannelSettingsSeq, i: int) -> rti.connextdds.ChannelSettings`

Remove and return the item at index i

remove (self: rti.connextdds.ChannelSettingsSeq, x: rti.connextdds.ChannelSettings) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class rti.connextdds.CharSeq

Bases: pybind11_object

__add__ (self: rti.connextdds.CharSeq, arg0: rti.connextdds.CharSeq) → rti.connextdds.CharSeq

__bool__ (*self*: rti.connexdds.CharSeq) → bool
 Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.CharSeq, *x*: str) → bool
 Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.CharSeq, *arg0*: int) -> None
 Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connexdds.CharSeq, *arg0*: slice) -> None
 Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.CharSeq, *arg0*: rti.connexdds.CharSeq) → bool

__getitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.CharSeq, *s*: slice) -> rti.connexdds.CharSeq
 Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.CharSeq, *arg0*: int) -> str

__getstate__ (*self*: rti.connexdds.CharSeq) → bytes

__hash__ = None

__iadd__ (*self*: rti.connexdds.CharSeq, *arg0*: rti.connexdds.CharSeq) → rti.connexdds.CharSeq

__imul__ (*self*: rti.connexdds.CharSeq, *arg0*: int) → rti.connexdds.CharSeq

__init__ (**args*, ***kwargs*)
 Overloaded function.

1. **__init__**(*self*: rti.connexdds.CharSeq, *arg0*: buffer) -> None
2. **__init__**(*self*: rti.connexdds.CharSeq) -> None
3. **__init__**(*self*: rti.connexdds.CharSeq, *arg0*: rti.connexdds.CharSeq) -> None
 Copy constructor
4. **__init__**(*self*: rti.connexdds.CharSeq, *arg0*: Iterable) -> None
5. **__init__**(*self*: rti.connexdds.CharSeq, *arg0*: int) -> None

__iter__ (*self*: rti.connexdds.CharSeq) → Iterator

__len__ (*self*: rti.connexdds.CharSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.CharSeq, *arg0*: int) → rti.connexdds.CharSeq

__ne__ (*self*: rti.connexdds.CharSeq, *arg0*: rti.connexdds.CharSeq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (*self*: rti.connexdds.CharSeq) → str

Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.CharSeq, *arg0*: int) → rti.connexdds.CharSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.CharSeq, *arg0*: int, *arg1*: str) -> None

2. **__setitem__**(*self*: rti.connexdds.CharSeq, *arg0*: slice, *arg1*: rti.connexdds.CharSeq) -> None

Assign list elements using a slice object

__setstate__ (*self*: rti.connexdds.CharSeq, *arg0*: bytes) → None

append (*self*: rti.connexdds.CharSeq, *x*: str) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.CharSeq) → None

Clear the contents

count (*self*: rti.connexdds.CharSeq, *x*: str) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.CharSeq, *L*: rti.connexdds.CharSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.CharSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.CharSeq, *i*: int, *x*: str) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.CharSeq) -> str

Remove and return the last item

2. **pop**(*self*: rti.connexdds.CharSeq, *i*: int) -> str

Remove and return the item at index *i*

remove (*self*: rti.connexdds.CharSeq, *x*: str) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

resize (*self*: rti.connexdds.CharSeq, *arg0*: int) → None

resizes the vector to the given size

rti.connexdds.**CharType**

alias of *Int8Type*

class rti.connexdds.**CoherentAccess**

Bases: pybind11_object

__enter__ (*self*: rti.connexdds.CoherentAccess) → *rti.connexdds.CoherentAccess*

Start a context managed coherent access block.

__exit__ (*self*: rti.connexdds.CoherentAccess, *arg0*: object, *arg1*: object, *arg2*: object) → None

End a context managed coherent access block.

__init__ (*self*: rti.connexdds.CoherentAccess, *subscriber*: rti.connexdds.Subscriber) → None

Creating a CoherentAccess object indicates that the application is about to access the data samples in any of the DataReader objects attached to the provided Subscriber.

__module__ = 'rti.connexdds'

end (*self*: rti.connexdds.CoherentAccess) → None

Explicitly indicate that the application has finished accessing the data samples in DataReader objects managed by the Subscriber.

class rti.connexdds.**CoherentSet**

Bases: pybind11_object

__enter__ (*self*: rti.connexdds.CoherentSet) → *rti.connexdds.CoherentSet*

Context manage the CoherentSet.

__exit__ (*self*: rti.connexdds.CoherentSet, *arg0*: object, *arg1*: object, *arg2*: object) → None

__init__ (*self*: rti.connexdds.CoherentSet, *publisher*: rti.connexdds.Publisher) → None

Creating a CoherentSet object indicates that the application will begin a coherent set of modifications using DataWriter objects attached to the Publisher.

__module__ = 'rti.connexdds'

end (*self*: rti.connexdds.CoherentSet) → None

Explicitly terminate a coherent set initiated by the CoherentSet constructor.

class rti.connexdds.**CoherentSetInfo**

Bases: pybind11_object

UNKNOWN = <rti.connexdds.CoherentSetInfo object>

__eq__ (*self*: rti.connexdds.CoherentSetInfo, *arg0*: rti.connexdds.CoherentSetInfo) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.CoherentSetInfo) -> None

Create a default CoherentSetInfo.

2. **__init__**(*self*: rti.connexdds.CoherentSetInfo, *group_guid*: rti.connexdds.Guid, *coherent_set_sequence_number*: rti.connexdds.SequenceNumber, *group_coherent_set_sequence_number*: rti.connexdds.SequenceNumber, *incomplete_coherent_set*: bool) -> None

Create a CoherentSetInfo with the specified parameters.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.CoherentSetInfo, *arg0*: rti.connexdds.CoherentSetInfo) → bool

Test for inequality.

__str__ (*self*: rti.connexdds.CoherentSetInfo) → str

Convert CoherentSetInfo to str.

property coherent_set_sequence_number

The coherent set sequence number that identifies a sample as part of a DataWriter coherent set.

property group_coherent_set_sequence_number

The group coherent set sequence number that identifies a sample as part of a group coherent set.

property group_guid

Coherent set group.

property incomplete_coherent_set

The incomplete coherent set status.

class rti.connexdds.CollectionType

Bases: *DynamicType*

__eq__ (*self*: rti.connexdds.CollectionType, *arg0*: rti.connexdds.CollectionType) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.CollectionType, *arg0*: rti.connexdds.CollectionType) → bool

Test for inequality.

property content_type

Gets the type of elements of this collection.

class `rti.connextdds.CompressionIdMask`

Bases: `pybind11_object`

ALL = `0000000000000111`

BZIP2 = `0000000000000010`

DEFAULT_PUBLICATION = `0000000000000000`

DEFAULT_SUBSCRIPTION = `0000000000000111`

LZ4 = `0000000000000100`

NONE = `0000000000000000`

ZLIB = `0000000000000001`

__and__ (*self*: `rti.connextdds.CompressionIdMask`, *arg0*: `rti.connextdds.CompressionIdMask`) → `rti.connextdds.CompressionIdMask`

Bitwise logical AND of masks.

__bool__ (*self*: `rti.connextdds.CompressionIdMask`) → `bool`

Test if any bits are set.

__contains__ (*self*: `rti.connextdds.CompressionIdMask`, *arg0*: `rti.connextdds.CompressionIdMask`) → `bool`

__eq__ (*self*: `rti.connextdds.CompressionIdMask`, *arg0*: `rti.connextdds.CompressionIdMask`) → `bool`

Compare masks for equality.

__getitem__ (*self*: `rti.connextdds.CompressionIdMask`, *arg0*: `int`) → `bool`

Get individual mask bit.

__hash__ = `None`

__iand__ (*self*: `rti.connextdds.CompressionIdMask`, *arg0*: `rti.connextdds.CompressionIdMask`) → `rti.connextdds.CompressionIdMask`

Set mask to logical AND with another mask.

__ilshift__ (*self*: `rti.connextdds.CompressionIdMask`, *arg0*: `int`) → `rti.connextdds.CompressionIdMask`

Left shift bits in mask.

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.CompressionIdMask) -> None`

Create a `CompressionIdMask` with no bits set.

2. `__init__(self: rti.connextdds.CompressionIdMask, value: int) -> None`

Creates a mask from the bits in an integer.

`__int__(self: rti.connextdds.CompressionIdMask) → int`

Convert mask to int.

`__ior__(self: rti.connextdds.CompressionIdMask, arg0: rti.connextdds.CompressionIdMask) → rti.connextdds.CompressionIdMask`

Set mask to logical OR with another mask.

`__irshift__(self: rti.connextdds.CompressionIdMask, arg0: int) → rti.connextdds.CompressionIdMask`

Right shift bits in mask.

`__ixor__(self: rti.connextdds.CompressionIdMask, arg0: rti.connextdds.CompressionIdMask) → rti.connextdds.CompressionIdMask`

Set mask to logical XOR with another mask.

`__lshift__(self: rti.connextdds.CompressionIdMask, arg0: int) → rti.connextdds.CompressionIdMask`

Left shift bits in mask.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.CompressionIdMask, arg0: rti.connextdds.CompressionIdMask) → bool`

Compare masks for inequality.

`__or__(self: rti.connextdds.CompressionIdMask, arg0: rti.connextdds.CompressionIdMask) → rti.connextdds.CompressionIdMask`

Bitwise logical OR of masks.

`__repr__(self: rti.connextdds.CompressionIdMask) → str`

Convert mask to string.

`__rshift__(self: rti.connextdds.CompressionIdMask, arg0: int) → rti.connextdds.CompressionIdMask`

Right shift bits in mask.

`__setitem__(self: rti.connextdds.CompressionIdMask, arg0: int, arg1: bool) → None`

Set individual mask bit

`__str__(self: rti.connextdds.CompressionIdMask) → str`

Convert mask to string.

`__xor__(self: rti.connextdds.CompressionIdMask, arg0: rti.connextdds.CompressionIdMask) → rti.connextdds.CompressionIdMask`

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (*args, **kwargs)

Overloaded function.

1. flip(self: rti.connexdds.CompressionIdMask) -> rti.connexdds.CompressionIdMask

Flip all bits in the mask.

2. flip(self: rti.connexdds.CompressionIdMask, pos: int) -> rti.connexdds.CompressionIdMask

Flip the mask bit at the specified position.

reset (*args, **kwargs)

Overloaded function.

1. reset(self: rti.connexdds.CompressionIdMask) -> rti.connexdds.CompressionIdMask

Clear all bits in the mask.

2. reset(self: rti.connexdds.CompressionIdMask, pos: int) -> rti.connexdds.CompressionIdMask

Clear the mask bit at the specified position.

set (*args, **kwargs)

Overloaded function.

1. set(self: rti.connexdds.CompressionIdMask) -> rti.connexdds.CompressionIdMask

Set all bits in the mask.

2. set(self: rti.connexdds.CompressionIdMask, pos: int, value: bool = True) -> rti.connexdds.CompressionIdMask

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (self: rti.connexdds.CompressionIdMask, pos: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (self: rti.connexdds.CompressionIdMask) → bool

Test if all bits are set.

test_any (self: rti.connexdds.CompressionIdMask) → bool

Test if any bits are set.

test_none (self: rti.connexdds.CompressionIdMask) → bool

Test if none of the bits are set.

class rti.connexdds.CompressionSettings

Bases: pybind11_object

```
COMPRESSION_LEVEL_BEST_COMPRESSION = 10
```

```
COMPRESSION_LEVEL_BEST_SPEED = 1
```

```
COMPRESSION_LEVEL_DEFAULT = 10
```

```
__eq__(self: rti.connexdds.CompressionSettings, arg0: rti.connexdds.CompressionSettings) → bool
```

Test for equality.

```
__hash__ = None
```

```
__init__(*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connexdds.CompressionSettings) -> None`

Create a CompressionSettings object with default settings.

2. `__init__(self: rti.connexdds.CompressionSettings, compression_ids: rti.connexdds.CompressionIdMask) -> None`

Creates an instance with the given compression_ids.

3. `__init__(self: rti.connexdds.CompressionSettings, compression_ids: rti.connexdds.CompressionIdMask, writer_compression_level: int, writer_compression_threshold: int) -> None`

Creates an instance with the given compression_ids, writer_compression_level and writer_compression_threshold.

```
__module__ = 'rti.connexdds'
```

```
__ne__(self: rti.connexdds.CompressionSettings, arg0: rti.connexdds.CompressionSettings) → bool
```

Test for inequality.

```
property compression_ids
```

Compression ID settings.

```
property writer_compression_level
```

Writer compression level.

```
property writer_compression_threshold
```

Writer compression threshold

```
class rti.connexdds.Condition
```

Bases: *ICondition*

```
__init__(*args, **kwargs)
```

```
__module__ = 'rti.connexdds'
```

```
class rti.connexdds.ConditionSeq
```

Bases: *pybind11_object*

__add__ (*self*: rti.connexdds.ConditionSeq, *arg0*: rti.connexdds.ConditionSeq) → *rti.connexdds.ConditionSeq*

__bool__ (*self*: rti.connexdds.ConditionSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.ConditionSeq, *x*: rti.connexdds.Condition) → bool
Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.ConditionSeq, *arg0*: int) -> None
Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connexdds.ConditionSeq, *arg0*: slice) -> None
Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.ConditionSeq, *arg0*: rti.connexdds.ConditionSeq) → bool

__getitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.ConditionSeq, *s*: slice) -> rti.connexdds.ConditionSeq
Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.ConditionSeq, *arg0*: int) -> rti.connexdds.Condition

__hash__ = None

__iadd__ (*self*: rti.connexdds.ConditionSeq, *arg0*: rti.connexdds.ConditionSeq) → *rti.connexdds.ConditionSeq*

__imul__ (*self*: rti.connexdds.ConditionSeq, *arg0*: int) → *rti.connexdds.ConditionSeq*

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: rti.connexdds.ConditionSeq) -> None
2. **__init__**(*self*: rti.connexdds.ConditionSeq, *arg0*: rti.connexdds.ConditionSeq) -> None
Copy constructor
3. **__init__**(*self*: rti.connexdds.ConditionSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.ConditionSeq) → Iterator

__len__ (*self*: rti.connexdds.ConditionSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.ConditionSeq, *arg0*: int) → *rti.connexdds.ConditionSeq*

__ne__ (*self*: rti.connexdds.ConditionSeq, *arg0*: rti.connexdds.ConditionSeq) → bool

__rmul__ (*self*: rti.connexdds.ConditionSeq, *arg0*: int) → rti.connexdds.ConditionSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.ConditionSeq, *arg0*: int, *arg1*: rti.connexdds.Condition) -> None
2. **__setitem__**(*self*: rti.connexdds.ConditionSeq, *arg0*: slice, *arg1*: rti.connexdds.ConditionSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.ConditionSeq, *x*: rti.connexdds.Condition) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.ConditionSeq) → None

Clear the contents

count (*self*: rti.connexdds.ConditionSeq, *x*: rti.connexdds.Condition) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.ConditionSeq, *L*: rti.connexdds.ConditionSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.ConditionSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.ConditionSeq, *i*: int, *x*: rti.connexdds.Condition) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.ConditionSeq) -> rti.connexdds.Condition

Remove and return the last item

2. **pop**(*self*: rti.connexdds.ConditionSeq, *i*: int) -> rti.connexdds.Condition

Remove and return the item at index *i*

remove (*self*: rti.connexdds.ConditionSeq, *x*: rti.connexdds.Condition) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.ContentFilterBase

Bases: pybind11_object

```
__init__ (*args, **kwargs)
```

```
__module__ = 'rti.connexdds'
```

```
class rti.connexdds.ContentFilterProperty
```

```
Bases: pybind11_object
```

```
__eq__ (self: rti.connexdds.ContentFilterProperty, arg0: rti.connexdds.ContentFilterProperty) → bool
```

Test for equality.

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.ContentFilterProperty, arg0: rti.connexdds.ContentFilterProperty) → bool
```

Test for inequality.

```
property content_filter_topic_name
```

The ContentFilteredTopic filter parameters.

```
property filter_class_name
```

Identifies the filter class this filter belongs to.

```
property filter_expression
```

The filter expression.

```
property related_topic_name
```

The name of the ContentFilteredTopic's related Topic.

```
class rti.connexdds.ContentFilteredTopic
```

```
Bases: ITopicDescription, IAnyTopic
```

```
__eq__ (self: rti.connexdds.ContentFilteredTopic, arg0: rti.connexdds.ContentFilteredTopic) → bool
```

Test for equality.

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connexdds.ContentFilteredTopic, topic: rti.connexdds.Topic, name: str, contentfilter: rti.connexdds.Filter) -> None`

Create a ContentFilteredTopic with a name and Filter.

2. `__init__(self: rti.connexdds.ContentFilteredTopic, topic_description: rti.connexdds.ITopicDescription) -> None`

Cast a TopicDescription to a ContentFilteredTopic.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ContentFilteredTopic, *arg0*: rti.connexdds.ContentFilteredTopic) → bool

Test for inequality.

append_to_expression_parameter (*self*: rti.connexdds.ContentFilteredTopic, *index*: int, *extension*: str) → None

Append the extension to the end of parameter at the provided index, separated by a comma.

property filter_expression

Get the filter expression

property filter_parameters

Get/set the filter parameters.

static find (*participant*: rti.connexdds.DomainParticipant, *name*: str) → Optional[rti.connexdds.ContentFilteredTopic]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

remove_from_expression_parameter (*self*: rti.connexdds.ContentFilteredTopic, *index*: int, *remove_term*: str) → None

Removes the specified term from the parameter at the provided index.

set_filter (*self*: rti.connexdds.ContentFilteredTopic, *arg0*: rti.connexdds.Filter) → None

Set the filter.

property topic

Get the underlying Topic.

class rti.connexdds.ContentFilteredTopicSeq

Bases: pybind11_object

__add__ (*self*: List[rti.connexdds.ContentFilteredTopic], *arg0*: List[rti.connexdds.ContentFilteredTopic]) → List[rti.connexdds.ContentFilteredTopic]

__bool__ (*self*: List[rti.connexdds.ContentFilteredTopic]) → bool

Check whether the list is nonempty

__contains__ (*self*: List[rti.connexdds.ContentFilteredTopic], *x*: rti.connexdds.ContentFilteredTopic) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: List[rti.connexdds.ContentFilteredTopic], *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: List[rti.connexdds.ContentFilteredTopic], *arg0*: slice) -> None

Delete list elements using a slice object

`__eq__` (*self*: List[rti.connexdds.ContentFilteredTopic], *arg0*: List[rti.connexdds.ContentFilteredTopic]) → bool

`__getitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__(self: List[rti.connexdds.ContentFilteredTopic], s: slice) -> List[rti.connexdds.ContentFilteredTopic]`

Retrieve list elements using a slice object

2. `__getitem__(self: List[rti.connexdds.ContentFilteredTopic], arg0: int) -> rti.connexdds.ContentFilteredTopic`

`__hash__` = None

`__iadd__` (*self*: List[rti.connexdds.ContentFilteredTopic], *arg0*: List[rti.connexdds.ContentFilteredTopic]) → List[rti.connexdds.ContentFilteredTopic]

`__imul__` (*self*: List[rti.connexdds.ContentFilteredTopic], *arg0*: int) → List[rti.connexdds.ContentFilteredTopic]

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.ContentFilteredTopicSeq) -> None`
2. `__init__(self: rti.connexdds.ContentFilteredTopicSeq, arg0: List[rti.connexdds.ContentFilteredTopic]) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.ContentFilteredTopicSeq, arg0: Iterable) -> None`

`__iter__` (*self*: List[rti.connexdds.ContentFilteredTopic]) → Iterator

`__len__` (*self*: List[rti.connexdds.ContentFilteredTopic]) → int

`__module__` = 'rti.connexdds'

`__mul__` (*self*: List[rti.connexdds.ContentFilteredTopic], *arg0*: int) → List[rti.connexdds.ContentFilteredTopic]

`__ne__` (*self*: List[rti.connexdds.ContentFilteredTopic], *arg0*: List[rti.connexdds.ContentFilteredTopic]) → bool

`__rmul__` (*self*: List[rti.connexdds.ContentFilteredTopic], *arg0*: int) → List[rti.connexdds.ContentFilteredTopic]

`__setitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__setitem__(self: List[rti.connexdds.ContentFilteredTopic], arg0: int, arg1: rti.connexdds.ContentFilteredTopic) -> None`

2. `__setitem__(self: List[rti.connextdds.ContentFilteredTopic], arg0: slice, arg1: List[rti.connextdds.ContentFilteredTopic]) -> None`

Assign list elements using a slice object

append (*self: List[rti.connextdds.ContentFilteredTopic]*, *x: rti.connextdds.ContentFilteredTopic*) → None

Add an item to the end of the list

clear (*self: List[rti.connextdds.ContentFilteredTopic]*) → None

Clear the contents

count (*self: List[rti.connextdds.ContentFilteredTopic]*, *x: rti.connextdds.ContentFilteredTopic*) → int

Return the number of times *x* appears in the list

extend (**args, **kwargs*)

Overloaded function.

1. `extend(self: List[rti.connextdds.ContentFilteredTopic], L: List[rti.connextdds.ContentFilteredTopic]) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: List[rti.connextdds.ContentFilteredTopic], L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self: List[rti.connextdds.ContentFilteredTopic]*, *i: int*, *x: rti.connextdds.ContentFilteredTopic*) → None

Insert an item at a given position.

pop (**args, **kwargs*)

Overloaded function.

1. `pop(self: List[rti.connextdds.ContentFilteredTopic]) -> rti.connextdds.ContentFilteredTopic`

Remove and return the last item

2. `pop(self: List[rti.connextdds.ContentFilteredTopic], i: int) -> rti.connextdds.ContentFilteredTopic`

Remove and return the item at index *i*

remove (*self: List[rti.connextdds.ContentFilteredTopic]*, *x: rti.connextdds.ContentFilteredTopic*) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class `rti.connextdds.Cookie`

Bases: `pybind11_object`

Unique identifier for a written data sample in the form of a sequence of bytes.

__eq__ (*self: rti.connextdds.Cookie*, *arg0: rti.connextdds.Cookie*) → bool

Compare Cookies for equality

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.Cookie) -> None

Creates an empty Cookie.

2. **__init__**(self: rti.connexdds.Cookie, bytes: rti.connexdds.ByteVector) -> None

Cookie from bytes.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.Cookie, arg0: rti.connexdds.Cookie) → bool

Compare Cookies for inequality

property value

Retrieve the bytes of the Cookie's contents.

class rti.connexdds.CookieSeq

Bases: pybind11_object

__add__ (self: rti.connexdds.CookieSeq, arg0: rti.connexdds.CookieSeq) → rti.connexdds.CookieSeq

__bool__ (self: rti.connexdds.CookieSeq) → bool

Check whether the list is nonempty

__contains__ (self: rti.connexdds.CookieSeq, x: rti.connexdds.Cookie) → bool

Return true the container contains x

__delitem__ (*args, **kwargs)

Overloaded function.

1. **__delitem__**(self: rti.connexdds.CookieSeq, arg0: int) -> None

Delete the list elements at index i

2. **__delitem__**(self: rti.connexdds.CookieSeq, arg0: slice) -> None

Delete list elements using a slice object

__eq__ (self: rti.connexdds.CookieSeq, arg0: rti.connexdds.CookieSeq) → bool

__getitem__ (*args, **kwargs)

Overloaded function.

1. **__getitem__**(self: rti.connexdds.CookieSeq, s: slice) -> rti.connexdds.CookieSeq

Retrieve list elements using a slice object

2. **__getitem__**(self: rti.connexdds.CookieSeq, arg0: int) -> rti.connexdds.Cookie

__hash__ = None

__iadd__ (*self*: rti.connexdds.CookieSeq, *arg0*: rti.connexdds.CookieSeq) → *rti.connexdds.CookieSeq*

__imul__ (*self*: rti.connexdds.CookieSeq, *arg0*: int) → *rti.connexdds.CookieSeq*

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: rti.connexdds.CookieSeq) -> None
2. **__init__**(*self*: rti.connexdds.CookieSeq, *arg0*: rti.connexdds.CookieSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.CookieSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.CookieSeq) → Iterator

__len__ (*self*: rti.connexdds.CookieSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.CookieSeq, *arg0*: int) → *rti.connexdds.CookieSeq*

__ne__ (*self*: rti.connexdds.CookieSeq, *arg0*: rti.connexdds.CookieSeq) → bool

__repr__ (*self*: rti.connexdds.CookieSeq) → str
Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.CookieSeq, *arg0*: int) → *rti.connexdds.CookieSeq*

__setitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.CookieSeq, *arg0*: int, *arg1*: rti.connexdds.Cookie) -> None
2. **__setitem__**(*self*: rti.connexdds.CookieSeq, *arg0*: slice, *arg1*: rti.connexdds.CookieSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.CookieSeq, *x*: rti.connexdds.Cookie) → None
Add an item to the end of the list

clear (*self*: rti.connexdds.CookieSeq) → None
Clear the contents

count (*self*: rti.connexdds.CookieSeq, *x*: rti.connexdds.Cookie) → int
Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)
Overloaded function.

1. **extend**(*self*: rti.connexdds.CookieSeq, *L*: rti.connexdds.CookieSeq) -> None

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.CookieSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.CookieSeq, *i*: int, *x*: rti.connextdds.Cookie) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connextdds.CookieSeq) -> rti.connextdds.Cookie`

Remove and return the last item

2. `pop(self: rti.connextdds.CookieSeq, i: int) -> rti.connextdds.Cookie`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.CookieSeq, *x*: rti.connextdds.Cookie) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connextdds.CookieVector

Bases: pybind11_object

A DDS standard container with functionality similar to a C++ vector.

__eq__ (*self*: rti.connextdds.CookieVector, *arg0*: rti.connextdds.CookieVector) → bool

Compare CookieVectors for equality.

__getitem__ (*self*: rti.connextdds.CookieVector, *arg0*: int) → rti.connextdds.Cookie

Get the value at the specified index.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.CookieVector) -> None`

Create an empty CookieVector

2. `__init__(self: rti.connextdds.CookieVector, size: int) -> None`

Create a CookieVector with a preallocated size.

3. `__init__(self: rti.connextdds.CookieVector, vector: rti.connextdds.CookieVector) -> None`

Create a copy from another CookieVector.

4. `__init__(self: rti.connextdds.CookieVector, arg0: Iterable) -> None`

5. `__init__(self: rti.connextdds.CookieVector, list: rti.connextdds.CookieSeq) -> None`

Create a CookieVector from a list of values.

__iter__ (*self*: rti.connextdds.CookieVector) → Iterator

Iterate over the contents of the vector.

__len__ (*self*: rti.connexdds.CookieVector) → int

Get the length of the CookieVector.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.CookieVector, *arg0*: rti.connexdds.CookieVector) → bool

Compare CookieVectors for inequality.

__setitem__ (*self*: rti.connexdds.CookieVector, *arg0*: int, *arg1*: rti.connexdds.Cookie) → None

Set the value at the specified index.

clear (*self*: rti.connexdds.CookieVector) → None

Resize CookieVector to 0.

resize (*self*: rti.connexdds.CookieVector, *size*: int) → None

Resize CookieVector.

class rti.connexdds.DataReader

Bases: *IDataReader*

class LoanedSample

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DataReader.LoanedSample) -> None

Basic constructor

2. **__init__**(*self*: rti.connexdds.DataReader.LoanedSample, *data*: rti.connexdds.User-DataSample, *info*: rti.connexdds.SampleInfo) -> None

Construct LoanedSample with data and info.

__iter__ (*self*: rti.connexdds.DataReader.LoanedSample) → object

__module__ = 'rti.connexdds'

property data

Get the data associated with the sample.

property info

Get the info associated with the sample.

class LoanedSamples

Bases: pybind11_object

__enter__ (*self*: rti.connexdds.DataReader.LoanedSamples) → *rti.connexdds.DataReader.LoanedSamples*

Enter a context for the loaned samples, loan returned on context exit.

__exit__ (*self*: rti.connexdds.DataReader.LoanedSamples, *arg0*: object, *arg1*: object, *arg2*: object) → None

Exit the context for the loaned samples, returning the resources.

__getitem__ (*self*: rti.connextdds.DataReader.LoanedSamples, *arg0*: int) → *rti.connextdds.DataReader.LoanedSample*

Access a LoanedSample object in an array-like syntax

__init__ (*self*: rti.connextdds.DataReader.LoanedSamples) → None

Create an empty LoanedSamples object.

__iter__ (*self*: rti.connextdds.DataReader.LoanedSamples) → Iterator

__len__ (*self*: rti.connextdds.DataReader.LoanedSamples) → int

Get the number of samples in the loan.

__module__ = 'rti.connextdds'

property length

Get the number of samples in the loan.

return_loan (*self*: rti.connextdds.DataReader.LoanedSamples) → None

Returns the loan to the DataReader.

class Selector

Bases: pybind11_object

__init__ (*self*: rti.connextdds.DataReader.Selector, *datareader*: rti.connextdds.DataReader) → None

Create a Selector for a DataReader to read/take based on selected conditions

__module__ = 'rti.connextdds'

condition (*self*: rti.connextdds.DataReader.Selector, *condition*: rti.connextdds.IReadCondition) → *rti.connextdds.DataReader.Selector*

Select samples based on a ReadCondition.

content (*self*: rti.connextdds.DataReader.Selector, *query*: rti.connextdds.Query) → *rti.connextdds.DataReader.Selector*

Select samples based on a Query.

instance (*self*: rti.connextdds.DataReader.Selector, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.DataReader.Selector*

Select a specific instance to read/take.

max_samples (*self*: rti.connextdds.DataReader.Selector, *max*: int) → *rti.connextdds.DataReader.Selector*

Limit the number of samples read/taken by the Select.

next_instance (*self*: rti.connextdds.DataReader.Selector, *previous*: rti.connextdds.InstanceHandle) → *rti.connextdds.DataReader.Selector*

Select the instance after the specified instance to read/take.

read (*self*: rti.connextdds.DataReader.Selector) → list

Read copies of all available data and info based on Selector settings.

read_data (*self*: rti.connexdds.DataReader.Selector) → list

Read copies of all available valid data based on Selector settings.

read_loaned (*self*: rti.connexdds.DataReader.Selector) →
rti.connexdds.DataReader.LoanedSamples

(Advanced) Read data as a collection of loaned data in C format and info objects based on Selector settings

state (*self*: rti.connexdds.DataReader.Selector, *state*: rti.connexdds.DataState) →
rti.connexdds.DataReader.Selector

Select samples with a specified data state.

take (*self*: rti.connexdds.DataReader.Selector) → list

Take copies of all available data and info based on Selector settings.

take_data (*self*: rti.connexdds.DataReader.Selector) → list

Take copies of all available valid data based on Selector settings.

take_loaned (*self*: rti.connexdds.DataReader.Selector) →
rti.connexdds.DataReader.LoanedSamples

(Advanced) Take data as a collection of loaned data in C format and info objects based on Selector settings

__enter__ (*self*: rti.connexdds.DataReader) → *rti.connexdds.DataReader*

Enter a context for this DataReader, to be cleaned up on exiting context

__eq__ (*self*: rti.connexdds.DataReader, *arg0*: rti.connexdds.DataReader) → bool

Test for equality.

__exit__ (*self*: rti.connexdds.DataReader, *arg0*: object, *arg1*: object, *arg2*: object) → None

Exit the context for this DataReader, cleaning up resources.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DataReader, *topic*: rti.connexdds.Topic) -> None

Create a DataReader in the implicit subscriber with default QoS.

2. **__init__**(*self*: rti.connexdds.DataReader, *sub*: rti.connexdds.Subscriber, *topic*: rti.connexdds.Topic) -> None

Create a DataReader in a subscriber with default QoS.

3. **__init__**(*self*: rti.connexdds.DataReader, *topic*: rti.connexdds.Topic, *qos*: rti.connexdds.DataReaderQos, *listener*: rti.connexdds.DataReaderListener = None, *mask*: rti.connexdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader in the implicit subscriber with specific QoS and a listener.

4. `__init__(self: rti.connextdds.DataReader, sub: rti.connextdds.Subscriber, topic: rti.connextdds.Topic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Create a DataReader in a subscriber with specific QoS and a listener.

5. `__init__(self: rti.connextdds.DataReader, cft: rti.connextdds.ContentFilteredTopic) -> None`

Create a DataReader with a ContentFilteredTopic in the participant's implicit subscriber with default QoS.

6. `__init__(self: rti.connextdds.DataReader, sub: rti.connextdds.Subscriber, cft: rti.connextdds.ContentFilteredTopic) -> None`

Create a DataReader with a ContentFilteredTopic in a subscriber with default QoS.

7. `__init__(self: rti.connextdds.DataReader, cft: rti.connextdds.ContentFilteredTopic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with specific QoS.

8. `__init__(self: rti.connextdds.DataReader, sub: rti.connextdds.Subscriber, cft: rti.connextdds.ContentFilteredTopic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Create a DataReader with a ContentFilteredTopic in a subscriber with specific QoS.

9. `__init__(self: rti.connextdds.DataReader, reader: rti.connextdds.IAnyDataReader) -> None`

Get a typed DataReader from an AnyDataReader.

10. `__init__(self: rti.connextdds.DataReader, entity: rti.connextdds.IEntity) -> None`

Get a typed DataReader from an Entity.

`__lshift__ (self: rti.connextdds.DataReader, arg0: rti.connextdds.DataReaderQos) → rti.connextdds.DataReader`

Set the DataReaderQos for this DataReader.

`__module__ = 'rti.connextdds'`

`__ne__ (self: rti.connextdds.DataReader, arg0: rti.connextdds.DataReader) → bool`
Test for inequality.

`__rshift__ (self: rti.connextdds.DataReader, arg0: rti.connextdds.DataReaderQos) → rti.connextdds.DataReader`

Get the DataReaderQos from this DataReader

`acknowledge_all (*args, **kwargs)`

Overloaded function.

1. `acknowledge_all(self: rti.connextdds.DataReader) -> None`

Acknowledge all previously accessed samples.

2. `acknowledge_all(self: rti.connextdds.DataReader, arg0: rti.connextdds.AckResponseData)`
-> None

Acknowledge all previously accessed samples.

acknowledge_sample (**args, **kwargs*)

Overloaded function.

1. `acknowledge_sample(self: rti.connextdds.DataReader, sample_info: rti.connextdds.SampleInfo)` -> None

Acknowledge a single sample.

2. `acknowledge_sample(self: rti.connextdds.DataReader, sample_info: rti.connextdds.SampleInfo, ack_response_data: rti.connextdds.AckResponseData)` -> None

Acknowledge a single sample with ack response data.

close (*self: rti.connextdds.DataReader*) → None

Close this DataReader.

property datareader_cache_status

Get the DataReaderCacheStatus for the DataReader.

property datareader_protocol_status

Get the DataReaderProtocolStatus for the DataReader.

property default_filter_state

Returns the filter state for the read/take operations.

static find_all_by_topic (*subscriber: rti.connextdds.Subscriber, topic_name: str*) →
List[*rti.connextdds.DataReader*]

Retrieve all DataReaders for the given topic name in the subscriber.

static find_by_name (**args, **kwargs*)

Overloaded function.

1. `find_by_name(participant: rti.connextdds.DomainParticipant, name: str)` -> Optional[*rti.connextdds.DataReader*]

Find DataReader in DomainParticipant with the DataReader's name, returning the first found.

2. `find_by_name(subscriber: rti.connextdds.Subscriber, name: str)` -> Optional[*rti.connextdds.DataReader*]

Find DataReader in Subscriber with the DataReader's name, returning the first found.

static find_by_topic (*subscriber: rti.connextdds.Subscriber, name: str*) →
Optional[*rti.connextdds.DataReader*]

Find DataReader in Subscriber with a topic name, returning the first found.

is_matched_publication_alive (*self: rti.connextdds.DataReader, arg0: rti.connextdds.InstanceHandle*) → bool

Check if a matched publication is alive.

key_value (*self*: rti.connextdds.DataReader, *handle*: rti.connextdds.InstanceHandle) → object
 Retrieve the instance key that corresponds to an instance handle.

property listener

Gets or sets the listener with StatusMask.ALL

property liveliness_changed_status

Get the LivelinessChangedStatus for this DataReader.

lookup_instance (*args, **kwargs)

Overloaded function.

1. lookup_instance(*self*: rti.connextdds.DataReader, *key_holder*: rti.connextdds.UserDataSample) -> rti.connextdds.InstanceHandle

Retrieve the instance handle that corresponds to an instance key_holder

2. lookup_instance(*self*: rti.connextdds.DataReader, *key_holder*: object) -> rti.connextdds.InstanceHandle

Retrieve the instance handle that corresponds to an instance key_holder

matched_publication_data (*self*: rti.connextdds.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.PublicationBuiltinTopicData*

Get the PublicationBuiltinTopicData for a publication matched to this DataReader.

matched_publication_datareader_protocol_status (*self*: rti.connextdds.DataReader, *publication_handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.DataReaderProtocolStatus*

Get the DataReaderProtocolStatus for the DataReader based on the matched publication handle.

matched_publication_participant_data (*self*: rti.connextdds.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a publication matched to this DataReader.

property matched_publications

Get a copy of the list of the currently matched publication handles.

property qos

The DataReaderQos for this DataReader.

This property's getter returns a deep copy.

read (*self*: rti.connextdds.DataReader) → list

Read copies of all available data and info

read_data (*self*: rti.connextdds.DataReader) → list
 Read copies of all available valid data

read_loaned (*self*: rti.connextdds.DataReader) → *rti.connextdds.DataReader.LoanedSamples*
 (Advanced) Read all data as a collection of loaned data in C format and info objects

property requested_deadline_missed_status
 Get the RequestedDeadlineMissed status for the DataReader

property requested_incompatible_qos_status
 Get the RequestedIncompatibleQosStatus for the DataReader.

property sample_lost_status
 Get the SampleLostStatus for the DataReader.

property sample_rejected_status
 Get the SampleRejectedStatus for the DataReader.

select (*self*: rti.connextdds.DataReader) → *rti.connextdds.DataReader.Selector*
 Get a Selector to perform complex data selections, such as per-instance selection, content, and status filtering.

set_listener (*self*: rti.connextdds.DataReader, *listener*: rti.connextdds.DataReaderListener, *event_mask*: rti.connextdds.StatusMask) → None
 Set the listener and associated event mask.

property subscriber
 Returns the parent Subscriber of the DataReader.

property subscription_matched_status
 Get the SubscriptionMatchedStatus for the DataReader.

take (*self*: rti.connextdds.DataReader) → list
 Take copies of all available data and info

take_async (*condition*: *Optional*[ReadCondition] = None)

take_data (*self*: rti.connextdds.DataReader) → list
 Take copies of all available valid data

take_data_async (*condition*: *Optional*[ReadCondition] = None)

take_loaned (*self*: rti.connextdds.DataReader) → *rti.connextdds.DataReader.LoanedSamples*
 (Advanced) Take all data as a collection of loaned data in C format and info objects

property topic_description
 Returns the TopicDescription associated with the DataReader.

property topic_name
 Get the topic name associated with this DataReader.

property type_name

Get the type name associated with this DataReader.

wait_for_historical_data (*self*: rti.connextdds.DataReader, *max_wait*: rti.connextdds.Duration) → None

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind.

wait_for_historical_data_async (*self*: rti.connextdds.DataReader, *max_wait*: rti.connextdds.Duration) → object

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind. This call is awaitable and only for use with asyncio.

class rti.connextdds.DataReaderCacheStatus

Bases: pybind11_object

__init__ (*args, **kwargs)

__module__ = 'rti.connextdds'

property alive_instance_count

The number of instances in the DataReader’s queue with an instance state equal to InstanceState.ALIVE.

property alive_instance_count_peak

The highest value of DataReaderCacheStatus.alive_instance_count over the lifetime of the DataReader.

property compressed_sample_count

The number of received compressed samples.

property content_filter_dropped_sample_count

The number of user samples filtered by the DataReader due to Content-Filtered Topics.

property detached_instance_count

The number of instances in the DataReader’s queue with an instance state equal to InstanceState.NOT_ALIVE_DISPOSED.

property detached_instance_count_peak

The highest value of DataReaderCacheStatus.detached_instance_count over the lifetime of the DataReader.

property disposed_instance_count

The number of instances in the DataReader’s queue with an instance state equal to InstanceState.NOT_ALIVE_DISPOSED.

property disposed_instance_count_peak

The number of minimal instance states currently being maintained in the DataReader’s queue.

property expired_dropped_sample_count

The number of samples expired by the DataReader due to Lifespan QoS or the autopurge sample delays.

property no_writers_instance_count

The number of instances in the DataReader's queue with an instance state equal to InstanceState.NOT_ALIVE_NO_WRITERS.

property no_writers_instance_count_peak

The highest value of DataReaderCacheStatus.no_writers_instance_count over the lifetime of the DataReader.

property old_source_timestamp_dropped_sample_count

The number of samples dropped as a result of receiving a sample older than the last one, using DestinationOrderKind.BY_SOURCE_TIMESTAMP.

property ownership_dropped_sample_count

The number of samples dropped as a result of receiving a sample from a DataWriter with a lower strength, using Exclusive Ownership.

property replaced_dropped_sample_count

The number of samples replaced by the DataReader due to HistoryKind.KEEP_LAST replacement.

property sample_count

The number of samples in the DataReader's queue.

property sample_count_peak

The highest number of samples in the DataReader's queue over the lifetime of the reader.

property time_based_filter_dropped_sample_count

The number of user samples filtered by the DataReader due to TimeBasedFilter QoS.

property tolerance_source_timestamp_dropped_sample_count

The number of samples dropped as a result of receiving a sample in the future, using DestinationOrderKind.BY_SOURCE_TIMESTAMP.

property total_samples_dropped_by_instance_replacement

The number of samples with sample state SampleState.NOT_READ that were dropped when removing an instance due to instance replacement. See DataReaderResourceLimits.instance_replacement for more details about when instances are replaced.

property virtual_duplicate_dropped_sample_count

The number of virtual duplicate samples dropped by the DataReader. A sample is a virtual duplicate if it has the same identity (Virtual Writer GUID and Virtual Sequence Number) as a previously received sample.

property writer_removed_batch_sample_dropped_sample_count

The number of batch samples received by the DataReader that were marked as removed by the DataWriter.

```
class rti.connexdds.DataReaderInstanceRemovalKind
```

```
Bases: pybind11_object
```

```
ANY_INSTANCE = <DataReaderInstanceRemovalKind.ANY_INSTANCE: 3>
```

class DataReaderInstanceRemovalKind

Bases: pybind11_object

Members:

NO_INSTANCE : No instance can be removed.

EMPTY_INSTANCES : Only empty instances can be removed

FULLY_PROCESSED_INSTANCES : Only fully-processed instances can be removed.

ANY_INSTANCE : Any instance can be removed.

ANY_INSTANCE = <DataReaderInstanceRemovalKind.ANY_INSTANCE: 3>**EMPTY_INSTANCES =****<DataReaderInstanceRemovalKind.EMPTY_INSTANCES: 1>****FULLY_PROCESSED_INSTANCES =****<DataReaderInstanceRemovalKind.FULLY_PROCESSED_INSTANCES: 2>****NO_INSTANCE = <DataReaderInstanceRemovalKind.NO_INSTANCE: 0>****__eq__** (*self: object, other: object*) → bool**__getstate__** (*self: object*) → int**__hash__** (*self: object*) → int**__index__** (*self: rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind*) → int**__init__** (*self: rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind, value: int*) → None**__int__** (*self: rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind*) → int**__members__ = {'ANY_INSTANCE':****<DataReaderInstanceRemovalKind.ANY_INSTANCE: 3>,****'EMPTY_INSTANCES':****<DataReaderInstanceRemovalKind.EMPTY_INSTANCES: 1>,****'FULLY_PROCESSED_INSTANCES':****<DataReaderInstanceRemovalKind.FULLY_PROCESSED_INSTANCES: 2>,****'NO_INSTANCE': <DataReaderInstanceRemovalKind.NO_INSTANCE: 0>}****__module__ = 'rti.connexdds'****__ne__** (*self: object, other: object*) → bool

__repr__ (*self: object*) → str

__setstate__ (*self: rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind, state: int*) → None

__str__ ()
name(*self: handle*) -> str

property name

property value

EMPTY_INSTANCES = <DataReaderInstanceRemovalKind.EMPTY_INSTANCES: 1>

FULLY_PROCESSED_INSTANCES =
<DataReaderInstanceRemovalKind.FULLY_PROCESSED_INSTANCES: 2>

NO_INSTANCE = <DataReaderInstanceRemovalKind.NO_INSTANCE: 0>

__eq__ (*self: rti.connextdds.DataReaderInstanceRemovalKind, arg0: rti.connextdds.DataReaderInstanceRemovalKind*) → bool

Apply operator to underlying enumerated values.

__ge__ (*self: rti.connextdds.DataReaderInstanceRemovalKind, arg0: rti.connextdds.DataReaderInstanceRemovalKind*) → bool

Apply operator to underlying enumerated values.

__gt__ (*self: rti.connextdds.DataReaderInstanceRemovalKind, arg0: rti.connextdds.DataReaderInstanceRemovalKind*) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args, **kwargs*)

Overloaded function.

1. **__init__**(*self: rti.connextdds.DataReaderInstanceRemovalKind*) -> None

Initializes enum to 0.

2. **__init__**(*self: rti.connextdds.DataReaderInstanceRemovalKind, arg0: rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind*) -> None

Copy constructor.

__int__ (*self: rti.connextdds.DataReaderInstanceRemovalKind*) → *rti.connextdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind*

Int conversion.

__le__ (*self*: rti.connexdds.DataReaderInstanceRemovalKind, *arg0*: rti.connexdds.DataReaderInstanceRemovalKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.DataReaderInstanceRemovalKind, *arg0*: rti.connexdds.DataReaderInstanceRemovalKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DataReaderInstanceRemovalKind, *arg0*: rti.connexdds.DataReaderInstanceRemovalKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.DataReaderInstanceRemovalKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.DataReaderListener

Bases: pybind11_object

__init__ (*self*: rti.connexdds.DataReaderListener) → None

__module__ = 'rti.connexdds'

on_data_available (*self*: rti.connexdds.DataReaderListener, *arg0*: rti.connexdds.DataReader) → None

Data available callback.

on_liveliness_changed (*self*: rti.connexdds.DataReaderListener, *arg0*: rti.connexdds.DataReader, *arg1*: rti.connexdds.LivelinessChangedStatus) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connexdds.DataReaderListener, *arg0*: rti.connexdds.DataReader, *arg1*: rti.connexdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connexdds.DataReaderListener, *arg0*: rti.connexdds.DataReader, *arg1*: rti.connexdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connexdds.DataReaderListener, *arg0*: rti.connexdds.DataReader, *arg1*: rti.connexdds.SampleLostStatus) → None

Sample lost callback.

on_sample_rejected (*self*: rti.connextdds.DataReaderListener, *arg0*: rti.connextdds.DataReader, *arg1*: rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connextdds.DataReaderListener, *arg0*: rti.connextdds.DataReader, *arg1*: rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class rti.connextdds.DataReaderProtocol

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.DataReaderProtocol, *arg0*: rti.connextdds.DataReaderProtocol) → bool
Compare for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.DataReaderProtocol) → None
Create a default DataReaderProtocol policy.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.DataReaderProtocol, *arg0*: rti.connextdds.DataReaderProtocol) → bool
Compare for inequality.

property disable_positive_acks

Get/set the boolean for whether the reader will use send positive acknowledgments.

property expects_inline_qos

Get/set the boolean for whether the reader will expect inline QoS with each sample.

property propagate_dispose_of_unregistered_instances

Get/set the boolean for whether an instance can move to the not_alive_disposed state without being in the alive state.

property propagate_unregister_of_disposed_instances

Get/set the boolean for whether an instance can move to the not_alive_no_writers state without being in the alive state.

property rtps_object_id

Get/set the RTPS object ID.

property rtps_reliable_reader

Get/set the reliable reader protocol settings.

property virtual_guid

Get/set a copy of the virtual GUID.

class rti.connextdds.DataReaderProtocolStatus

Bases: pybind11_object

Information about the DataReader's protocol status.

```
__init__ (*args, **kwargs)
```

```
__module__ = 'rti.connextdds'
```

```
property dropped_fragment_count
```

The number of DATA_FRAG messages that have been dropped by a DataReader.

```
property duplicate_sample_bytes
```

The number of bytes of sample data from a remote DataWriter received, not for the first time, by a local DataReader.

```
property duplicate_sample_count
```

The number of samples from a remote DataWriter received, not for the first time, by a local DataReader.

```
property first_available_sample_sequence_number
```

Sequence number of the first available sample in a matched DataWriters reliability queue.

```
property last_available_sample_sequence_number
```

Sequence number of the last available sample in a matched Datawriter's reliability queue.

```
property last_committed_sample_sequence_number
```

Sequence number of the newest sample received from the matched DataWriter committed to the DataReader's queue.

```
property out_of_range_rejected_sample_count
```

The number of samples dropped by the DataReader due to received window is full and the sample is out-of-order.

```
property reassembled_sample_count
```

The number of fragmented samples that have been reassembled by a DataReader.

```
property received_fragment_count
```

The number of DATA_FRAG messages that have been received by this DataReader.

```
property received_gap_bytes
```

The number of bytes of GAP data received from remote DataWriter to this DataReader.

```
property received_gap_count
```

The number of GAPS received from remote DataWriter to this DataReader.

```
property received_heartbeat_bytes
```

The number of bytes of Heartbeat data from a remote DataWriter received by a local DataReader.

```
property received_heartbeat_count
```

The number of Heartbeats from a remote DataWriter received by a local DataReader.

```
property received_sample_bytes
```

The number of bytes from samples received by a DataReader.

```
property received_sample_count
```

The number of samples received by a DataReader.

property rejected_sample_count

The number of times samples were rejected due to exceptions in the receive path.

property sent_ack_bytes

The number of bytes of ACK data sent from a local DataReader to a matching remote DataWriter.

property sent_ack_count

The number of ACKs sent from a local DataReader to a matching remote DataWriter.

property sent_nack_bytes

The number of bytes of NACK data sent from a local DataReader to a matching remote DataWriter.

property sent_nack_count

The number of NACKs sent from a local DataReader to a matching remote DataWriter.

property sent_nack_fragment_bytes

The number of NACK fragment bytes that have been sent from a DataReader to a DataWriter.

property sent_nack_fragment_count

The number of NACK fragments that have been sent from a DataReader to a DataWriter.

property uncommitted_sample_count

Number of received samples that are not yet available to be read or taken, due to being received out of order.

class rti.connextdds.DataReaderQos

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.DataReaderQos, *arg0*: rti.connextdds.DataReaderQos) → bool

Test for equality

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.DataReaderQos) -> None

Create a DataReaderQos with the default value for each policy.

2. **__init__**(*self*: rti.connextdds.DataReaderQos, *reader*: rti.connextdds.IAnyDataReader) -> None

Create a DataReaderQos with settings equivalent to those of the provided DataReader.

3. **__init__**(*self*: rti.connextdds.DataReaderQos, *other*: rti.connextdds.DataReaderQos) -> None

Create a copy of a DataReaderQos object.

__lshift__ (**args*, ***kwargs*)

Overloaded function.

1. **__lshift__**(*self*: rti.connextdds.DataReaderQos, *arg0*: rti.connextdds.Durability) -> rti.connextdds.DataReaderQos

Set the DurabilityQoS.

2. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Deadline) -> rti.connextdds.DataReaderQos`

Set the DeadlineQoS.

3. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.LatencyBudget) -> rti.connextdds.DataReaderQos`

Set the LatencyBudgetQoS.

4. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Liveliness) -> rti.connextdds.DataReaderQos`

Set the LivelinessQoS.

5. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Reliability) -> rti.connextdds.DataReaderQos`

Set the ReliabilityQoS.

6. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DestinationOrder) -> rti.connextdds.DataReaderQos`

Set the DestinationOrderQoS.

7. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.History) -> rti.connextdds.DataReaderQos`

Set the HistoryQoS.

8. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.ResourceLimits) -> rti.connextdds.DataReaderQos`

Set the ResourceLimitsQoS.

9. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.UserData) -> rti.connextdds.DataReaderQos`

Set the UserDataQoS.

10. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Ownership) -> rti.connextdds.DataReaderQos`

Set the OwnershipQoS.

11. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TimeBasedFilter) -> rti.connextdds.DataReaderQos`

Set the TimeBasedFilterQoS.

12. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.ReaderDataLifecycle) -> rti.connextdds.DataReaderQos`

Set the ReaderDataLifecycleQoS.

13. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TransportPriority) -> rti.connextdds.DataReaderQos`

Set the TransportPriorityQoS.

14. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TypeConsistencyEnforcement) -> rti.connextdds.DataReaderQos`

Set the TypeConsistencyEnforcementQoS.

15. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DataReaderResourceLimits) -> rti.connextdds.DataReaderQos`

Set the DataReaderResourceLimitsQoS.

16. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DataReaderProtocol) -> rti.connextdds.DataReaderQos`

Set the DataReaderProtocolQoS.

17. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TransportSelection) -> rti.connextdds.DataReaderQos`

Set the TransportSelectionQoS.

18. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TransportUnicast) -> rti.connextdds.DataReaderQos`

Set the TransportUnicastQoS.

19. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TransportMulticast) -> rti.connextdds.DataReaderQos`

Set the TransportMulticastQoS.

20. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Property) -> rti.connextdds.DataReaderQos`

Set the PropertyQoS.

21. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Service) -> rti.connextdds.DataReaderQos`

Set the ServiceQoS.

22. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Availability) -> rti.connextdds.DataReaderQos`

Set the AvailabilityQoS.

23. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.EntityName) -> rti.connextdds.DataReaderQos`

Set the EntityNameQoS.

24. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TypeSupport) -> rti.connextdds.DataReaderQos`

Set the TypeSupportQoS.

25. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DataRepresentation) -> rti.connextdds.DataReaderQos`

Set the DataRepresentationQoS.

26. `__lshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DataTag) -> rti.connextdds.DataReaderQos`

Set the DataTagQoS.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DataReaderQos) -> bool`

Test for inequality.

`__rshift__(*args, **kwargs)`

Overloaded function.

1. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Durability) -> rti.connextdds.DataReaderQos`

Get the DurabilityQoS.

2. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Deadline) -> rti.connextdds.DataReaderQos`

Get the DeadlineQoS.

3. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.LatencyBudget) -> rti.connextdds.DataReaderQos`

Get the LatencyBudgetQoS.

4. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Liveliness) -> rti.connextdds.DataReaderQos`

Get the LivelinessQoS.

5. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Reliability) -> rti.connextdds.DataReaderQos`

Get the ReliabilityQoS.

6. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DestinationOrder) -> rti.connextdds.DataReaderQos`

Get the DestinationOrderQoS.

7. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.History) -> rti.connextdds.DataReaderQos`

Get the HistoryQoS.

8. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.ResourceLimits) -> rti.connextdds.DataReaderQos`

Get the ResourceLimitsQoS.

9. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.UserData) -> rti.connextdds.DataReaderQos`

Get the UserDataQoS.

10. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Ownership) -> rti.connextdds.DataReaderQos`

Get the OwnershipQoS.

11. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TimeBasedFilter) -> rti.connextdds.DataReaderQos`

Get the TimeBasedFilterQoS.

12. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.ReaderDataLifecycle) -> rti.connextdds.DataReaderQos`

Get the ReaderDataLifecycleQoS.

13. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TransportPriority) -> rti.connextdds.DataReaderQos`

Get the TransportPriorityQoS.

14. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TypeConsistencyEnforcement) -> rti.connextdds.DataReaderQos`

Get the TypeConsistencyEnforcementQoS.

15. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DataReaderResourceLimits) -> rti.connextdds.DataReaderQos`

Get the DataReaderResourceLimitsQoS.

16. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DataReaderProtocol) -> rti.connextdds.DataReaderQos`

Get the DataReaderProtocolQoS.

17. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TransportSelection) -> rti.connextdds.DataReaderQos`

Get the TransportSelectionQoS.

18. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TransportUnicast) -> rti.connextdds.DataReaderQos`

Get the TransportUnicastQoS.

19. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TransportMulticast) -> rti.connextdds.DataReaderQos`

Get the TransportMulticastQoS.

20. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Property) -> rti.connextdds.DataReaderQos`

Get the PropertyQoS.

21. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Service) -> rti.connextdds.DataReaderQos`

Get the ServiceQoS.

22. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.Availability) -> rti.connextdds.DataReaderQos`

Get the AvailabilityQoS.

23. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.EntityName) -> rti.connextdds.DataReaderQos`

Get the EntityNameQoS.

24. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.TypeSupport) -> rti.connextdds.DataReaderQos`

Get the TypeSupportQoS.

25. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DataRepresentation) -> rti.connextdds.DataReaderQos`

Get the DataRepresentationQoS.

26. `__rshift__(self: rti.connextdds.DataReaderQos, arg0: rti.connextdds.DataTag) -> rti.connextdds.DataReaderQos`

Get the DataTagQoS.

`__str__(self: rti.connextdds.DataReaderQos) -> str`

property availability

Get/set Availability QoS.

property data_reader_protocol

(Deprecated) use protocol instead

property data_reader_resource_limits

(Deprecated) use reader_resource_limits instead

property data_representation

Get/set DataRepresentation QoS.

property data_tag

Get/set DataTag QoS.

property deadline

Get/set Deadline QoS.

property destination_order

Get/set DestinationOrder QoS.

property durability

Get/set Durability QoS.

property entity_name

Get/set EntityName QoS.

property history

Get/set History QoS.

property latency_budget

Get/set LatencyBudget QoS.

property liveliness

Get/set Liveliness QoS.

property ownership

Get/set Ownership QoS.

property property

Get/set Property QoS.

property protocol

Get/set DataReaderProtocol QoS.

property reader_data_lifecycle

Get/set ReaderDataLifecycle QoS.

property reader_resource_limits

Get/set DataReaderResourceLimits QoS.

property reliability

Get/set Reliability QoS.

property resource_limits

Get/set ResourceLimits QoS.

property service

Get/set Service QoS.

property time_based_filter

Get/set TimeBasedFilter QoS.

to_string (*self*: rti.connexdds.DataReaderQos, *format*: rti.connexdds.QosPrintFormat = *QosPrintFormat()*, *base*: *Optional*[rti.connexdds.DataReaderQos] = *None*, *print_all*: *bool* = *False*) → str

Convert QoS to string based on params.

property transport_multicast

Get/set TransportMulticast QoS.

property transport_priority

Get/set TransportPriority QoS.

property transport_selection

Get/set TransportSelection QoS.

property transport_unicast

Get/set TransportUnicast QoS.

property type_consistency
Get/set TypeConsistencyEnforcement QoS.

property type_consistency_enforcement
(Deprecated) use type_consistency instead

property type_support
Get/set TypeSupport QoS.

property user_data
Get/set UserData QoS.

class rti.connexdds.DataReaderResourceLimits

Bases: pybind11_object

AUTO_MAX_TOTAL_INSTANCES = 0

__eq__ (*self*: rti.connexdds.DataReaderResourceLimits, *arg0*: rti.connexdds.DataReaderResourceLimits) → bool

__hash__ = None

__init__ (*self*: rti.connexdds.DataReaderResourceLimits) → None
Create a default DataReaderResourceLimits policy with default settings.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DataReaderResourceLimits, *arg0*: rti.connexdds.DataReaderResourceLimits) → bool

property autopurge_remote_not_alive_writer_delay

Maximum duration for which the DataReader will maintain information regarding a DataWriter once the DataWriter has become not alive.

This property's getter returns a deep copy.

property autopurge_remote_virtual_writer_delay

Maximum duration for which the DataReader will maintain information regarding a virtual DataWriter once it has become not alive.

This property's getter returns a deep copy.

property disable_fragmentation_support

Determines whether the DataReader can receive fragmented samples.

property dynamically_allocate_fragmented_samples

Determines whether the DataReader pre-allocates storage for storing fragmented samples.

property initial_fragmented_samples

The initial number of samples for which a DataReader may store fragments.

property initial_infos

The initial number of info units that a DataReader can use to store SampleInfo.

property initial_outstanding_reads

The initial number of outstanding calls to read/take (or one of their variants) on the same DataReader for which memory has not been returned by calling `LoanedSamples.return_loan()`.

property initial_remote_virtual_writers

The initial number of remote virtual writers from which a DataReader may read, including all instances.

property initial_remote_virtual_writers_per_instance

The initial number of virtual remote writers that can be associated with an instance.

property initial_remote_writers

The initial number of remote writers from which a DataReader may read, including all instances.

property initial_remote_writers_per_instance

The initial number of remote writers from which a DataReader may read a single instance.

property initial_topic_queries

The initial number of TopicQueries allocated by a DataReader.

property instance_replacement

The instance replacement policy.

property keep_minimum_state_for_instances

Whether or not keep a minimum instance state for up to `DataReaderResourceLimits.max_total_instances`.

property max_app_ack_response_length

Maximum length of application-level acknowledgment response data.

property max_fragmented_samples

The maximum number of samples for which the DataReader may store fragments at a given point in time.

property max_fragmented_samples_per_remote_writer

The maximum number of samples per remote writer for which a DataReader may store fragments.

property max_fragments_per_sample

Maximum number of fragments for a single sample.

property max_infos

The maximum number of info units that a DataReader can use to store `SampleInfo`.

property max_outstanding_reads

The max number of outstanding calls to read/take (or one of their variants) on the same DataReader for which memory has not been returned by calling `LoanedSamples.return_loan()`.

property max_query_condition_filters

The maximum number of query condition filters a reader is allowed.

property max_remote_virtual_writers

The maximum number of remote virtual writers from which a DataReader may read, including all instances.

property max_remote_virtual_writers_per_instance

The maximum number of virtual remote writers that can be associated with an instance.

property max_remote_writers

The maximum number of remote writers from which a DataReader may read, including all instances.

property max_remote_writers_per_instance

The maximum number of remote writers from which a DataReader may read a single instance.

property max_remote_writers_per_sample

The maximum number of remote writers allowed to write the same sample.

property max_samples_per_read

The maximum number of data samples that the application can receive from the middleware in a single call to DataReader.read() or DataReader.take(). If more data exists in the middleware, the application will need to issue multiple read/take calls.

property max_samples_per_remote_writer

The maximum number of out-of-order samples from a given remote DataWriter that a DataReader may store when maintaining a reliable connection to the DataWriter.

property max_topic_queries

The maximum number of TopicQueries allocated by a DataReader.

property max_total_instances

Maximum number of instances for which a DataReader will keep state.

property shm_ref_transfer_mode_attached_segment_allocation

The initial number of TopicQueries allocated by a DataReader.

class

`rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings`, *arg0*: `rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings`) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings) -> None`

Creates an instance with the default removal kind for each instance state.

2. `__init__(self: rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings, alive_instance_removal: rti.connextdds.DataReaderInstanceRemovalKind, disposed_instance_removal: rti.connextdds.DataReaderInstanceRemovalKind, no_writers_instance_removal: rti.connextdds.DataReaderInstanceRemovalKind) -> None`

Creates an instance with the given `DataReaderInstanceRemovalKind` for alive, disposed and no_writers instances, respectively.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings, arg0: rti.connextdds.DataReaderResourceLimitsInstanceReplacementSettings) → bool`

Test for inequality.

property alive_instance_removal

The instance replacment policy for alive instances.

property disposed_instance_removal

The instance replacment policy for disposed instances.

property no_writers_instance_removal

The instance replacment policy for not-alive-no-writer instances.

class `rti.connextdds.DataReaderSeq`

Bases: `pybind11_object`

`__add__(self: List[rti.connextdds.DataReader], arg0: List[rti.connextdds.DataReader]) → List[rti.connextdds.DataReader]`

`__bool__(self: List[rti.connextdds.DataReader]) → bool`

Check whether the list is nonempty

`__contains__(self: List[rti.connextdds.DataReader], x: rti.connextdds.DataReader) → bool`

Return true the container contains `x`

`__delitem__(*args, **kwargs)`

Overloaded function.

1. `__delitem__(self: List[rti.connextdds.DataReader], arg0: int) -> None`

Delete the list elements at index `i`

2. `__delitem__(self: List[rti.connextdds.DataReader], arg0: slice) -> None`

Delete list elements using a slice object

`__eq__(self: List[rti.connextdds.DataReader], arg0: List[rti.connextdds.DataReader]) → bool`

`__getitem__(*args, **kwargs)`

Overloaded function.

1. `__getitem__(self: List[rti.connextdds.DataReader], s: slice) -> List[rti.connextdds.DataReader]`

Retrieve list elements using a slice object

2. `__getitem__(self: List[rti.connexdds.DataReader], arg0: int) -> rti.connexdds.DataReader`

`__hash__ = None`

`__iadd__(self: List[rti.connexdds.DataReader], arg0: List[rti.connexdds.DataReader]) -> List[rti.connexdds.DataReader]`

`__imul__(self: List[rti.connexdds.DataReader], arg0: int) -> List[rti.connexdds.DataReader]`

`__init__(*args, **kwargs)`
Overloaded function.

1. `__init__(self: rti.connexdds.DataReaderSeq) -> None`
2. `__init__(self: rti.connexdds.DataReaderSeq, arg0: List[rti.connexdds.DataReader]) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.DataReaderSeq, arg0: Iterable) -> None`

`__iter__(self: List[rti.connexdds.DataReader]) -> Iterator`

`__len__(self: List[rti.connexdds.DataReader]) -> int`

`__module__ = 'rti.connexdds'`

`__mul__(self: List[rti.connexdds.DataReader], arg0: int) -> List[rti.connexdds.DataReader]`

`__ne__(self: List[rti.connexdds.DataReader], arg0: List[rti.connexdds.DataReader]) -> bool`

`__rmul__(self: List[rti.connexdds.DataReader], arg0: int) -> List[rti.connexdds.DataReader]`

`__setitem__(*args, **kwargs)`
Overloaded function.

1. `__setitem__(self: List[rti.connexdds.DataReader], arg0: int, arg1: rti.connexdds.DataReader) -> None`
2. `__setitem__(self: List[rti.connexdds.DataReader], arg0: slice, arg1: List[rti.connexdds.DataReader]) -> None`

Assign list elements using a slice object

`append(self: List[rti.connexdds.DataReader], x: rti.connexdds.DataReader) -> None`
Add an item to the end of the list

`clear(self: List[rti.connexdds.DataReader]) -> None`
Clear the contents

`count(self: List[rti.connexdds.DataReader], x: rti.connexdds.DataReader) -> int`
Return the number of times `x` appears in the list

extend (*args, **kwargs)

Overloaded function.

1. extend(self: List[rti.connexdds.DataReader], L: List[rti.connexdds.DataReader]) -> None

Extend the list by appending all the items in the given list

2. extend(self: List[rti.connexdds.DataReader], L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (self: List[rti.connexdds.DataReader], i: int, x: rti.connexdds.DataReader) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: List[rti.connexdds.DataReader]) -> rti.connexdds.DataReader

Remove and return the last item

2. pop(self: List[rti.connexdds.DataReader], i: int) -> rti.connexdds.DataReader

Remove and return the item at index i

remove (self: List[rti.connexdds.DataReader], x: rti.connexdds.DataReader) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class rti.connexdds.DataRepresentation

Bases: pybind11_object

AUTO_ID = -1

XCDR = 0

XCDR2 = 2

XML = 1

__eq__ (self: rti.connexdds.DataRepresentation, arg0: rti.connexdds.DataRepresentation) → bool

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. __init__(self: rti.connexdds.DataRepresentation) -> None

Creates an instance with one element, AUTO_ID.

2. __init__(self: rti.connexdds.DataRepresentation, representations: rti.connexdds.Int16Seq) -> None

Creates an instance with a sequence of Data Representation IDs.

__module__ = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.DataRepresentation, *arg0*: rti.connexdds.DataRepresentation) → bool

property compression_settings

The compression settings.

property value

A sequence of IDs representing the allowed data representations.

This property uses copy semantics. Changes to the sequence are not reflected in the policy unless the sequence is committed back to the policy object via the property's setter.

class rti.connexdds.DataState

Bases: pybind11_object

`__eq__` (*self*: rti.connexdds.DataState, *arg0*: rti.connexdds.DataState) → bool

Compare DataState objects for equality.

`__hash__` = None

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.DataState) -> None`

Create a DataState with InstanceState.any(), ViewState.any(), and SampleState.ANY

2. `__init__(self: rti.connexdds.DataState, sample_state: rti.connexdds.SampleState) -> None`

Create a DataState with InstanceState.any(), ViewState.any(), and the provided SampleState.

3. `__init__(self: rti.connexdds.DataState, view_state: rti.connexdds.ViewState) -> None`

Create a DataState with InstanceState.any(), SampleState.any(), and the provided ViewState.

4. `__init__(self: rti.connexdds.DataState, instance_state: rti.connexdds.InstanceState) -> None`

Create a DataState with ViewState.any(), SampleState.any(), and the provided InstanceState.

5. `__init__(self: rti.connexdds.DataState, sample_state: rti.connexdds.SampleState, view_state: rti.connexdds.ViewState, instance_state: rti.connexdds.InstanceState) -> None`

Create a DataState with the provided SampleState, ViewState, and InstanceState.

`__lshift__` (**args*, ***kwargs*)

Overloaded function.

1. `__lshift__(self: rti.connexdds.DataState, arg0: rti.connexdds.SampleState) -> rti.connexdds.DataState`

Set the SampleState for the DataState.

2. `__lshift__(self: rti.connexdds.DataState, arg0: rti.connexdds.ViewState) -> rti.connexdds.DataState`

Set the ViewState for the DataState.

3. `__lshift__(self: rti.connextdds.DataState, arg0: rti.connextdds.InstanceState) -> rti.connextdds.DataState`

Set the InstanceState for the DataState.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.DataState, arg0: rti.connextdds.DataState) -> bool`

Compare DataState objects for inequality.

`__repr__(self: rti.connextdds.DataState) -> str`

`any = [sample_state = any, view_state = any, instance_state = any]`

`any_data = [sample_state = any, view_state = any, instance_state = alive]`

property instance_state

The InstanceState of the DataState.

`new_data = [sample_state = not_read, view_state = any, instance_state = alive]`

`new_instance = [sample_state = any, view_state = new_view, instance_state = alive]`

property sample_state

The SampleState of the DataState.

property view_state

The ViewState of the DataState.

class rti.connextdds.DataStateEx

Bases: `pybind11_object`

`__eq__(self: rti.connextdds.DataStateEx, arg0: rti.connextdds.DataStateEx) -> bool`

Compare DataStateEx objects for equality.

`__hash__ = None`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.DataStateEx) -> None`

Create a DataStateEx with InstanceState.ANY, ViewState.ANY, SampleState.ANY, and StreamKind.ANY.

2. `__init__(self: rti.connextdds.DataStateEx, stream_kind: rti.connextdds.StreamKind) -> None`

Create a DataStateEx with InstanceState.ANY, ViewState.ANY, SampleState.ANY and the provided StreamKind.

3. `__init__(self: rti.connextdds.DataStateEx, data_state: rti.connextdds.DataState) -> None`

Create a DataStateEx with the provided SampleState, ViewState, InstanceState, and StreamKind.ANY

4. `__init__(self: rti.connextdds.DataStateEx, data_state: rti.connextdds.DataState, stream_kind: rti.connextdds.StreamKind) -> None`

Create a DataStateEx with the provided SampleState, ViewState, InstanceState, and StreamKind.

`__lshift__ (*args, **kwargs)`

Overloaded function.

1. `__lshift__(self: rti.connextdds.DataStateEx, arg0: rti.connextdds.StreamKind) -> rti.connextdds.DataStateEx`

Set the StreamKind for the DataStateEx.

2. `__lshift__(self: rti.connextdds.DataStateEx, arg0: rti.connextdds.SampleState) -> rti.connextdds.DataState`

Set the SampleState for the DataState.

3. `__lshift__(self: rti.connextdds.DataStateEx, arg0: rti.connextdds.ViewState) -> rti.connextdds.DataState`

Set the ViewState for the DataState.

4. `__lshift__(self: rti.connextdds.DataStateEx, arg0: rti.connextdds.InstanceState) -> rti.connextdds.DataState`

Set the InstanceState for the DataState.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.DataStateEx, arg0: rti.connextdds.DataStateEx) → bool`

Compare DataStateEx objects for inequality.

`any = [sample_state = any, view_state = any, instance_state = any]`

`any_data = [sample_state = any, view_state = any, instance_state = alive]`

property data_state

Access the view, sample and instance states.

property instance_state

The InstanceState of the DataStateEx.

`new_data = [sample_state = not_read, view_state = any, instance_state = alive]`

`new_instance = [sample_state = any, view_state = new_view, instance_state = alive]`

property sample_state

The SampleState of the DataStateEx.

property stream_kind

The StreamKind.

property view_state

The ViewState of the DataStateEx.

class `rti.connexdds.DataTag`

Bases: `pybind11_object`

__contains__ (*self*: `rti.connexdds.DataTag`, *arg0*: `str`) → `bool`

__eq__ (*self*: `rti.connexdds.DataTag`, *arg0*: `rti.connexdds.DataTag`) → `bool`

Test for equality.

__getitem__ (*self*: `rti.connexdds.DataTag`, *arg0*: `str`) → `Optional[str]`

__hash__ = `None`

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.DataTag) -> None`

Creates a policy with an empty sequence of tags.

2. `__init__(self: rti.connexdds.DataTag, entries: rti.connexdds.StringPairSeq) -> None`

Adds tags from a list.

3. `__init__(self: rti.connexdds.DataTag, entries: dict) -> None`

Adds tags from a dictionary.

__len__ (*self*: `rti.connexdds.DataTag`) → `int`

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.DataTag`, *arg0*: `rti.connexdds.DataTag`) → `bool`

Test for inequality.

__setitem__ (*self*: `rti.connexdds.DataTag`, *arg0*: `str`, *arg1*: `str`) → `None`

exists (*self*: `rti.connexdds.DataTag`, *key*: `str`) → `bool`

Returns true if a tag exists.

get (*self*: `rti.connexdds.DataTag`, *key*: `str`) → `str`

Returns the value of a tag identified by a key if it exists.

get_all (*self*: `rti.connexdds.DataTag`) → `rti.connexdds.StringMap`

Retrieves a copy of all the entries.

remove (*self*: `rti.connexdds.DataTag`, *key*: `str`) → `bool`

Removes the tag identified by a key.

set (*args, **kwargs)

Overloaded function.

1. set(self: rti.connextdds.DataTag, entry_list: rti.connextdds.StringPairSeq) -> None

Adds or assigns tags from a list of string pairs.

2. set(self: rti.connextdds.DataTag, entry_map: dict) -> None

Adds or assigns tags from a dictionary.

3. set(self: rti.connextdds.DataTag, entry: Tuple[str, str]) -> None

Adds or assigns a tag from a pair of strings.

4. set(self: rti.connextdds.DataTag, key: str, value: str) -> None

Adds or assigns a tag from a key string and a value string.

size (self: rti.connextdds.DataTag) → int

Returns the number of tags.

try_get (self: rti.connextdds.DataTag, key: str) → Optional[str]

Returns the value of a tag identified by a key if it exists.

class rti.connextdds.DataWriter

Bases: *IEntity, IAnyDataWriter*

__enter__ (self: rti.connextdds.DataWriter) → *rti.connextdds.DataWriter*

Enter a context for this DataWriter, to be cleaned up on exiting context

__eq__ (self: rti.connextdds.DataWriter, arg0: rti.connextdds.DataWriter) → bool

Test for equality.

__exit__ (self: rti.connextdds.DataWriter, arg0: object, arg1: object, arg2: object) → None

Exit the context for this DataWriter, cleaning up resources.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. __init__(self: rti.connextdds.DataWriter, topic: rti.connextdds.Topic) -> None

Creates a DataWriter in the implicit publisher with default QoS.

2. __init__(self: rti.connextdds.DataWriter, topic: rti.connextdds.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.DataWriterListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Creates a DataWriter in the implicit publisher with specific QoS and optionally a listener.

3. __init__(self: rti.connextdds.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.Topic) -> None

Creates a DataWriter in a publisher with default QoS.

4. `__init__(self: rti.connextdds.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.DataWriterListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Creates a DataWriter in a publisher with specific QoS and optionally a listener.

5. `__init__(self: rti.connextdds.DataWriter, writer: rti.connextdds.IAnyDataWriter) -> None`

Create a typed DataWriter from an AnyDataWriter.

6. `__init__(self: rti.connextdds.DataWriter, entity: rti.connextdds.IEntity) -> None`

Create a typed DataWriter from an Entity.

`__lshift__ (*args, **kwargs)`

Overloaded function.

1. `__lshift__(self: rti.connextdds.DataWriter, arg0: rti.connextdds.DataWriterQos) -> rti.connextdds.DataWriter`

Sets the DataWriterQos.

2. `__lshift__(self: rti.connextdds.DataWriter, arg0: Tuple[object, rti.connextdds.Time]) -> rti.connextdds.DataWriter`

Writes a paired sample with a timestamp.

3. `__lshift__(self: rti.connextdds.DataWriter, arg0: Tuple[object, rti.connextdds.InstanceHandle]) -> rti.connextdds.DataWriter`

Writes a paired sample with an instance handle.

4. `__lshift__(self: rti.connextdds.DataWriter, arg0: List[Tuple[object, rti.connextdds.Time]]) -> rti.connextdds.DataWriter`

Writes a sequence of pairs of samples with timestamps.

5. `__lshift__(self: rti.connextdds.DataWriter, arg0: List[object]) -> rti.connextdds.DataWriter`

Writes a sequence of samples.

6. `__lshift__(self: rti.connextdds.DataWriter, arg0: object) -> rti.connextdds.DataWriter`

Writes a sample.

`__module__ = 'rti.connextdds'`

`__ne__ (self: rti.connextdds.DataWriter, arg0: rti.connextdds.DataWriter) -> bool`

Test for inequality.

`__rshift__ (self: rti.connextdds.DataWriter, arg0: rti.connextdds.DataWriterQos) -> rti.connextdds.DataWriter`

Get the DataWriterQos.

`assert_liveliness (self: rti.connextdds.DataWriter) -> None`

Manually asserts the liveliness of the DataWriter.

close (*self*: rti.connextdds.DataWriter) → None

Close this DataWriter.

property datawriter_cache_status

Get a copy of the cache status for this writer.

property datawriter_protocol_status

Get a copy of the protocol status for this writer.

dispose_instance (**args*, ***kwargs*)

Overloaded function.

1. `dispose_instance(self: rti.connextdds.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.DataWriter`

Dispose an instance.

2. `dispose_instance(self: rti.connextdds.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.DataWriter`

Dispose an instance with a timestamp.

3. `dispose_instance(self: rti.connextdds.DataWriter, params: rti.connextdds.WriteParams) -> None`

Dispose an instance with params.

static find_all_by_topic (*publisher*: rti.connextdds.Publisher, *topic_name*: str) → List[rti.connextdds.DataWriter]

Retrieve all DataWriters for the given topic name in the publisher.

static find_by_name (**args*, ***kwargs*)

Overloaded function.

1. `find_by_name(participant: rti.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.DataWriter]`

Find DataWriter in DomainParticipant with the provided name, returning the first found.

2. `find_by_name(publisher: rti.connextdds.Publisher, name: str) -> Optional[rti.connextdds.DataWriter]`

Find DataWriter in Publisher with the DataReader's name, returning the first found.

static find_by_topic (*publisher*: rti.connextdds.Publisher, *name*: str) → Optional[rti.connextdds.DataWriter]

Find DataWriter in publisher with a topic name, returning the first found.

flush (*self*: rti.connextdds.DataWriter) → None

Flushes the batch in progress in the context of the calling thread.

is_matched_subscription_active (*self*: rti.connextdds.DataWriter, *arg0*: rti.connextdds.InstanceHandle) → bool

A boolean indicating whether or not the matched subscription is active.

is_sample_app_acknowledged (*self*: rti.connextdds.DataWriter, *sample_id*: rti.connextdds.SampleIdentity) → bool

Indicates if a sample is considered application-acknowledged.

key_value (*self*: rti.connextdds.DataWriter, *handle*: rti.connextdds.InstanceHandle) → object

Retrieve the instance key that corresponds to an instance handle.

property listener

Get the listener associated with the DataWriter or set the listener.

property liveliness_lost_status

Get a copy of the LivelinessLostStatus.

lookup_instance (*self*: rti.connextdds.DataWriter, *key_holder*: object) → rti.connextdds.InstanceHandle

Retrieve the instance handle that corresponds to an instance key_holder

matched_subscription_data (*self*: rti.connextdds.DataWriter, *handle*: rti.connextdds.InstanceHandle) → rti.connextdds.SubscriptionBuiltinTopicData

Get the SubscriptionBuiltinTopicData for a subscription matched to this DataWriter.

matched_subscription_datawriter_protocol_status (*args, **kwargs)

Overloaded function.

1. matched_subscription_datawriter_protocol_status(*self*: rti.connextdds.DataWriter, *handle*: rti.connextdds.InstanceHandle) -> rti.connextdds.DataWriterProtocolStatus

Get a copy of the protocol status for this writer per a matched subscription handle.

2. matched_subscription_datawriter_protocol_status(*self*: rti.connextdds.DataWriter, *locator*: rti.connextdds.Locator) -> rti.connextdds.DataWriterProtocolStatus

Get a copy of the protocol status for this writer per a matched subscription locator.

matched_subscription_participant_data (*self*: rti.connextdds.DataWriter, *handle*: rti.connextdds.InstanceHandle) → rti.connextdds.ParticipantBuiltinTopicData

Get the ParticipantBuiltinTopicData for a subscription matched to this DataWriter.

property matched_subscriptions

Get a copy of the list of the currently matched subscription handles.

property matched_subscriptions_locators

The locators used to communicate with matched DataReaders.

property offered_deadline_missed_status

Get a copy of the OfferedDeadlineMissedStatus.

property offered_incompatible_qos_status

Get a copy of the OfferedIncompatibleQosStatus

property publication_matched_status

Get a copy of the PublicationMatchedStatus

property publisher

Get the Publisher that owns this DataWriter.

property qos

The DataWriterQos for this DataWriter. This property's getter returns a deep copy.

register_instance (**args, **kwargs*)

Overloaded function.

1. `register_instance(self: rti.connextdds.DataWriter, key_holder: object) -> rti.connextdds.InstanceHandle`

Informs RTI Connex that the application will be modifying a particular instance.

2. `register_instance(self: rti.connextdds.DataWriter, key_holder: object, timestamp: rti.connextdds.Time) -> rti.connextdds.InstanceHandle`

Informs RTI Connex that the application will be modifying a particular instance and specified the timestamp.

3. `register_instance(self: rti.connextdds.DataWriter, key_holder: object, params: rti.connextdds.WriteParams) -> rti.connextdds.InstanceHandle`

Registers instance with parameters.

property reliable_reader_activity_changed_status

Get a copy of the reliable reader activity changed status for this writer.

property reliable_writer_cache_changed_status

Get a copy of the reliable cache status for this writer.

property service_request_accepted_status

Get a copy of the service request accepted status for this writer.

set_listener (*self: rti.connextdds.DataWriter, listener: rti.connextdds.DataWriterListener, event_mask: rti.connextdds.StatusMask*) → None

Set the listener and event mask for the DataWriter.

property topic

Get the Topic object associated with this DataWriter.

property topic_name

Get the topic name associated with this DataWriter.

property type_name

Get the type name for the topic object associated with this DataWriter.

unregister_instance (**args, **kwargs*)

Overloaded function.

1. `unregister_instance(self: rti.connextdds.DataWriter, handle: rti.connextdds.InstanceHandle)`
`-> rti.connextdds.DataWriter`

Unregister an instance.

2. `unregister_instance(self: rti.connextdds.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time)` `-> rti.connextdds.DataWriter`

Unregister an instance with timestamp.

3. `unregister_instance(self: rti.connextdds.DataWriter, params: rti.connextdds.WriteParams)` `-> None`

Unregister an instance with parameters.

wait_for_acknowledgments (*self: rti.connextdds.DataWriter, max_wait: rti.connextdds.Duration*) `→ None`

Blocks the calling thread until all data written by a reliable `DataWriter` is acknowledged or until the timeout expires.

wait_for_asynchronous_publishing (*self: rti.connextdds.DataWriter, max_wait: rti.connextdds.Duration*) `→ None`

This operation blocks the calling thread (up to `max_wait`) until all data written by the asynchronous `DataWriter` is sent and acknowledged (if reliable) by all matched `DataReader` entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that `max_wait` elapsed before all the data was sent and/or acknowledged.

In other words, this guarantees that sending to best effort `DataReader` is complete in addition to what `DataWriter.wait_for_acknowledgments()` provides.

If the `DataWriter` does not have `PublishMode` kind set to `PublishModeKind.ASYNCHRONOUS` the operation will complete immediately

wait_for_asynchronous_publishing_async (*self: rti.connextdds.DataWriter, max_wait: rti.connextdds.Duration*) `→ object`

This function is awaitable until either a timeout of `max_wait` or all data written by the asynchronous `DataWriter` is sent and acknowledged (if reliable) by all matched `DataReader` entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that `max_wait` elapsed before all the data was sent and/or acknowledged. This function works with `asyncio`.

In other words, this guarantees that sending to best effort `DataReader` is complete in addition to what `DataWriter.wait_for_acknowledgments()` provides.

If the `DataWriter` does not have `PublishMode` kind set to `PublishModeKind.ASYNCHRONOUS` the operation will complete immediately

write (**args, **kwargs*)

Overloaded function.

1. `write(self: rti.connextdds.DataWriter, samples: List[object])` `-> None`

Write a sequence of samples.

2. write(self: rti.connextdds.DataWriter, samples: List[object], timestamp: rti.connextdds.Time) -> None

Write a sequence of samples with a timestamp.

3. write(self: rti.connextdds.DataWriter, sample: object) -> None

Write a sample.

4. write(self: rti.connextdds.DataWriter, sample: object, timestamp: rti.connextdds.Time) -> None

Write a sample with a specified timestamp.

5. write(self: rti.connextdds.DataWriter, sample: object, handle: rti.connextdds.InstanceHandle) -> None

Write a sample with an instance handle.

6. write(self: rti.connextdds.DataWriter, sample: object, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> None

Write a sample with an instance handle and specified timestamp.

7. write(self: rti.connextdds.DataWriter, sample: object, params: rti.connextdds.WriteParams) -> None

Write with advanced parameters.

```
class rti.connextdds.DataWriterCacheStatus
```

```
Bases: pybind11_object
```

```
__init__ (*args, **kwargs)
```

```
__module__ = 'rti.connextdds'
```

```
property alive_instance_count
```

Number of alive instances in the DataWriter's queue.

```
property alive_instance_count_peak
```

Highest number of alive instances in the writer's queue over the lifetime of the writer.

```
property disposed_instance_count
```

Number of disposed instances in the DataWriter's queue.

```
property disposed_instance_count_peak
```

Highest number of disposed instances in the writer's queue over the lifetime of the writer.

```
property sample_count
```

Number of samples in the DataWriter's queue, including unregister and dispose samples.

```
property sample_count_peak
```

Highest number of samples in the writer's queue over the lifetime of the writer.

```
property unregistered_instance_count
```

Number of unregistered instances in the DataWriter's queue.

property unregistered_instance_count_peak

Highest number of unregistered instances in the writer's queue over the lifetime of the writer.

class rti.connextdds.DataWriterListener

Bases: pybind11_object

__init__ (*self*: rti.connextdds.DataWriterListener) → None

__module__ = 'rti.connextdds'

on_application_acknowledgment (*self*: rti.connextdds.DataWriterListener, *arg0*: rti.connextdds.DataWriter, *arg1*: rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

on_instance_replaced (*self*: rti.connextdds.DataWriterListener, *arg0*: rti.connextdds.DataWriter, *arg1*: rti.connextdds.InstanceHandle) → None

On instance replaced callback.

on_liveliness_lost (*self*: rti.connextdds.DataWriterListener, *arg0*: rti.connextdds.DataWriter, *arg1*: rti.connextdds.LivelinessLostStatus) → None

Liveliness lost callback.

on_offered_deadline_missed (*self*: rti.connextdds.DataWriterListener, *arg0*: rti.connextdds.DataWriter, *arg1*: rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

on_offered_incompatible_qos (*self*: rti.connextdds.DataWriterListener, *arg0*: rti.connextdds.DataWriter, *arg1*: rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

on_publication_matched (*self*: rti.connextdds.DataWriterListener, *arg0*: rti.connextdds.DataWriter, *arg1*: rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connextdds.DataWriterListener, *arg0*: rti.connextdds.DataWriter, *arg1*: rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connextdds.DataWriterListener, *arg0*: rti.connextdds.DataWriter, *arg1*: rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connexdds.DataWriterListener, *arg0*: rti.connexdds.DataWriter, *arg1*: rti.connexdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class rti.connexdds.DataWriterProtocol

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.DataWriterProtocol, *arg0*: rti.connexdds.DataWriterProtocol) → bool

__hash__ = None

__init__ (*self*: rti.connexdds.DataWriterProtocol) → None

Create a default DataWriterProtocol policy.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DataWriterProtocol, *arg0*: rti.connexdds.DataWriterProtocol) → bool

property disable_inline_keyhash

Controls whether a keyhash is propagated on the wire with each sample.

property disable_positive_acks

Controls whether the DataWriter expects positive acknowledgments from matched DataReaders.

property initial_virtual_sequence_number

The initial virtual sequence number of the DataWriter.

property propagate_app_ack_with_no_response

Controls whether or not a DataWriter receives on_application_acknowledgment notifications with an empty or invalid response.

property push_on_write

Determines whether a sample is pushed on a call to write.

property rtps_object_id

The RTPS object ID of the DataWriter.

property rtps_reliable_writer

RTPS protocol-related configuration settings for the RTPS reliable writer associated to a DataWriter. This parameter only has effect if both the writer and the matching reader are configured with ReliabilityKind.RELIABLE.

property serialize_key_with_dispose

Controls whether or not the serialized key is propagated on the wire with dispose samples.

property virtual_guid

The virtual GUID of the DataWriter. This property uses value semantics.

```

class rti.connexdds.DataWriterProtocolStatus
    Bases: pybind11_object
    Information about the DataWriter protocol status
    __init__ (*args, **kwargs)
    __module__ = 'rti.connexdds'

    property first_available_sample_sequence_number
        The sequence number of the first available sample currently queued in the local DataWriter.

    property first_available_sample_virtual_sequence_number
        The virtual sequence number of the first available sample currently queued in the local DataWriter.

    property first_unacknowledged_sample_sequence_number
        The sequence number of the first unacknowledged sample currently queued in the local DataWriter.

    property first_unacknowledged_sample_subscription_handle
        The handle of a remote DataReader that has not acknowledged the first unacknowledged sample
        of the local DataWriter.

    property first_unacknowledged_sample_virtual_sequence_number
        The virtual sequence number of the first unacknowledged sample currently queued in the local
        DataWriter.

    property first_unelapsed_keep_duration_sample_sequence_number
        The sequence number of the first sample whose keep duration has not yet elapsed.

    property last_available_sample_sequence_number
        The sequence number of the last available sample currently queued in the local DataWriter.

    property last_available_sample_virtual_sequence_number
        The virtual sequence number of the last available sample currently queued in the local DataWriter.

    property pulled_fragment_bytes
        The number of bytes of DATA_FRAG messages that have been pulled from this DataWriter.

    property pulled_fragment_count
        The number of DATA_FRAG messages that have been pulled from this DataWriter.

    property pulled_sample_bytes
        The number of bytes of user samples pulled from local DataWriter by matching DataReaders.

    property pulled_sample_count
        The number of user samples pulled from local DataWriter by matching DataReaders.

    property pushed_fragment_bytes
        The number of bytes of DATA_FRAG messages that have been pushed by this DataWriter.

    property pushed_fragment_count
        The number of DATA_FRAG messages that have been pushed by this DataWriter.

```

property pushed_sample_bytes

The number of bytes of user samples pushed on write from a local DataWriter to a matching remote DataReader.

property pushed_sample_count

The number of user samples pushed on write from a local DataWriter to a matching remote DataReader.

property received_ack_bytes

The number of bytes of ACKs from a remote DataReader received by a local DataWriter.

property received_ack_count

The number of ACKs from a remote DataReader received by a local DataWriter.

property received_nack_bytes

The number of bytes of NACKs from a remote DataReader received by a local DataWriter.

property received_nack_count

The number of NACKs from a remote DataReader received by a local DataWriter.

property received_nack_fragment_bytes

The number of bytes of NACK_FRAG messages that have been received by this DataWriter.

property received_nack_fragment_count

The number of NACK_FRAG messages that have been received by this DataWriter.

property rejected_sample_count

The number of times a sample is rejected due to exceptions in the send path.

property send_window_size

Current maximum number of outstanding samples allowed in the DataWriter's queue.

property sent_gap_bytes

The number of bytes of GAPs sent from local DataWriter to matching remote DataReaders.

property sent_gap_count

The number of GAPs sent from local DataWriter to matching remote DataReaders.

property sent_heartbeat_bytes

The number of bytes of Heartbeats sent between a local DataWriter and matching remote DataReader.

property sent_heartbeat_count

The number of Heartbeats sent between a local DataWriter and matching remote DataReader.

class rti.connextdds.DataWriterQos

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.DataWriterQos, *arg0*: rti.connextdds.DataWriterQos) → bool

Test for equality

`__hash__ = None`

`__init__ (*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.DataWriterQos) -> None`

Create a DataWriterQos with the default value for each policy.

2. `__init__(self: rti.connextdds.DataWriterQos, writer: rti.connextdds.IAnyDataWriter) -> None`

Create a DataWriterQos with settings equivalent to those of the provided DataWriter.

3. `__init__(self: rti.connextdds.DataWriterQos, other: rti.connextdds.DataWriterQos) -> None`

Create a copy of a DataWriterQos object.

`__lshift__ (*args, **kwargs)`

Overloaded function.

1. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Durability) -> rti.connextdds.DataWriterQos`

Set the DurabilityQoS.

2. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DurabilityService) -> rti.connextdds.DataWriterQos`

Set the DurabilityServiceQoS.

3. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Deadline) -> rti.connextdds.DataWriterQos`

Set the DeadlineQoS.

4. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.LatencyBudget) -> rti.connextdds.DataWriterQos`

Set the LatencyBudgetQoS.

5. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Liveliness) -> rti.connextdds.DataWriterQos`

Set the LivelinessQoS.

6. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Reliability) -> rti.connextdds.DataWriterQos`

Set the ReliabilityQoS.

7. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DestinationOrder) -> rti.connextdds.DataWriterQos`

Set the DestinationOrderQoS.

8. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.History) -> rti.connextdds.DataWriterQos`

Set the HistoryQoS.

9. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.ResourceLimits) -> rti.connextdds.DataWriterQos`

Set the ResourceLimitsQoS.

10. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.TransportPriority) -> rti.connextdds.DataWriterQos`

Set the TransportPriorityQoS.

11. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Lifespan) -> rti.connextdds.DataWriterQos`

Set the LifespanQoS.

12. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.UserData) -> rti.connextdds.DataWriterQos`

Set the UserDataQoS.

13. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Ownership) -> rti.connextdds.DataWriterQos`

Set the OwnershipQoS.

14. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.OwnershipStrength) -> rti.connextdds.DataWriterQos`

Set the OwnershipStrengthQoS.

15. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.WriterDataLifecycle) -> rti.connextdds.DataWriterQos`

Set the WriterDataLifecycleQoS.

16. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DataWriterResourceLimits) -> rti.connextdds.DataWriterQos`

Set the DataWriterResourceLimitsQoS.

17. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DataWriterProtocol) -> rti.connextdds.DataWriterQos`

Set the DataWriterProtocolQoS.

18. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.TransportSelection) -> rti.connextdds.DataWriterQos`

Set the TransportSelectionQoS.

19. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.TransportUnicast) -> rti.connextdds.DataWriterQos`

Set the TransportUnicastQoS.

20. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.PublishMode) -> rti.connextdds.DataWriterQos`

Set the PublishModeQoS.

21. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Property) -> rti.connextdds.DataWriterQos`

Set the PropertyQoS.

22. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Service) -> rti.connextdds.DataWriterQos`

Set the ServiceQoS.

23. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Batch) -> rti.connextdds.DataWriterQos`

Set the BatchQoS.

24. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.MultiChannel) -> rti.connextdds.DataWriterQos`

Set the MultiChannelQoS.

25. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Availability) -> rti.connextdds.DataWriterQos`

Set the AvailabilityQoS.

26. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.EntityName) -> rti.connextdds.DataWriterQos`

Set the EntityNameQoS.

27. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.TopicQueryDispatch) -> rti.connextdds.DataWriterQos`

Set the TopicQueryDispatchQoS.

28. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.TypeSupport) -> rti.connextdds.DataWriterQos`

Set the TypeSupportQoS.

29. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DataRepresentation) -> rti.connextdds.DataWriterQos`

Set the DataRepresentationQoS.

30. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DataTag) -> rti.connextdds.DataWriterQos`

Set the DataTagQoS.

31. `__lshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DataWriterTransferMode) -> rti.connextdds.DataWriterQos`

Set the DataWriterTransferModeQoS.

```
__module__ = 'rti.connextdds'
```

`__ne__` (*self*: rti.connexdds.DataWriterQos, *arg0*: rti.connexdds.DataWriterQos) → bool

Test for inequality.

`__rshift__` (**args*, ***kwargs*)

Overloaded function.

1. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.Durability) -> rti.connexdds.DataWriterQos`

Get the DurabilityQoS.

2. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.DurabilityService) -> rti.connexdds.DataWriterQos`

Get the DurabilityServiceQoS.

3. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.Deadline) -> rti.connexdds.DataWriterQos`

Get the DeadlineQoS.

4. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.LatencyBudget) -> rti.connexdds.DataWriterQos`

Get the LatencyBudgetQoS.

5. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.Liveliness) -> rti.connexdds.DataWriterQos`

Get the LivelinessQoS.

6. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.Reliability) -> rti.connexdds.DataWriterQos`

Get the ReliabilityQoS.

7. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.DestinationOrder) -> rti.connexdds.DataWriterQos`

Get the DestinationOrderQoS.

8. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.History) -> rti.connexdds.DataWriterQos`

Get the HistoryQoS.

9. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.ResourceLimits) -> rti.connexdds.DataWriterQos`

Get the ResourceLimitsQoS.

10. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.TransportPriority) -> rti.connexdds.DataWriterQos`

Get the TransportPriorityQoS.

11. `__rshift__(self: rti.connexdds.DataWriterQos, arg0: rti.connexdds.Lifespan) -> rti.connexdds.DataWriterQos`

Get the LifespanQoS.

12. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.UserData) -> rti.connextdds.DataWriterQos`

Get the UserDataQoS.

13. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Ownership) -> rti.connextdds.DataWriterQos`

Get the OwnershipQoS.

14. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.OwnershipStrength) -> rti.connextdds.DataWriterQos`

Get the OwnershipStrengthQoS.

15. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.WriterDataLifecycle) -> rti.connextdds.DataWriterQos`

Get the WriterDataLifecycleQoS.

16. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DataWriterResourceLimits) -> rti.connextdds.DataWriterQos`

Get the DataWriterResourceLimitsQoS.

17. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DataWriterProtocol) -> rti.connextdds.DataWriterQos`

Get the DataWriterProtocolQoS.

18. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.TransportSelection) -> rti.connextdds.DataWriterQos`

Get the TransportSelectionQoS.

19. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.TransportUnicast) -> rti.connextdds.DataWriterQos`

Get the TransportUnicastQoS.

20. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.PublishMode) -> rti.connextdds.DataWriterQos`

Get the PublishModeQoS.

21. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Property) -> rti.connextdds.DataWriterQos`

Get the PropertyQoS.

22. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Service) -> rti.connextdds.DataWriterQos`

Get the ServiceQoS.

23. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Batch) -> rti.connextdds.DataWriterQos`

Get the BatchQoS.

24. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.MultiChannel) -> rti.connextdds.DataWriterQos`

Get the MultiChannelQoS.

25. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.Availability) -> rti.connextdds.DataWriterQos`

Get the AvailabilityQoS.

26. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.EntityName) -> rti.connextdds.DataWriterQos`

Get the EntityNameQoS.

27. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.TopicQueryDispatch) -> rti.connextdds.DataWriterQos`

Get the TopicQueryDispatchQoS.

28. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.TypeSupport) -> rti.connextdds.DataWriterQos`

Get the TypeSupportQoS.

29. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DataRepresentation) -> rti.connextdds.DataWriterQos`

Get the DataRepresentationQoS.

30. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DataTag) -> rti.connextdds.DataWriterQos`

Get the DataTagQoS.

31. `__rshift__(self: rti.connextdds.DataWriterQos, arg0: rti.connextdds.DataWriterTransferMode) -> rti.connextdds.DataWriterQos`

Get the DataWriterTransferModeQoS.

`__str__ (self: rti.connextdds.DataWriterQos) → str`

property availability

Get/set Availability QoS.

property batch

Get/set Batch QoS.

property data_representation

Get/set DataRepresentation QoS.

property data_tag

Get/set DataTag QoS.

property data_writer_protocol

(Deprecated) use protocol instead

property data_writer_resource_limits

(Deprecated) use writer_resource_limits instead

property data_writer_transfer_mode

Get/set DataWriterTransferMode QoS.

property deadline

Get/set Deadline QoS.

property destination_order

Get/set DestinationOrder QoS.

property durability

Get/set Durability QoS.

property durability_service

Get/set DurabilityService QoS.

property entity_name

Get/set EntityName QoS.

property history

Get/set History QoS.

property latency_budget

Get/set LatencyBudget QoS.

property lifespan

Get/set Lifespan QoS.

property liveliness

Get/set Liveliness QoS.

property multi_channel

Get/set MultiChannel QoS.

property ownership

Get/set Ownership QoS.

property ownership_strength

Get/set OwnershipStrength QoS.

property property

Get/set Property QoS.

property protocol

Get/set DataWriterProtocol QoS.

property publish_mode

Get/set PublishMode QoS.

property reliability

Get/set Reliability QoS.

property resource_limits

Get/set ResourceLimits QoS.

property service

Get/set Service QoS.

to_string (*self*: rti.connexdds.DataWriterQos, *format*: rti.connexdds.QosPrintFormat = *QosPrintFormat()*, *base*: *Optional*[rti.connexdds.DataWriterQos] = *None*, *print_all*: *bool* = *False*) → str

Convert QoS to string based on params.

property topic_query_dispatch

Get/set TopicQueryDispatch QoS.

property transport_priority

Get/set TransportPriority QoS.

property transport_selection

Get/set TransportSelection QoS.

property transport_unicast

Get/set TransportUnicast QoS.

property type_support

Get/set TypeSupport QoS.

property user_data

Get/set UserData QoS.

property writer_data_lifecycle

Get/set WriterDataLifecycle QoS.

property writer_resource_limits

Get/set DataWriterResourceLimits QoS.

class rti.connexdds.DataWriterResourceLimits

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.DataWriterResourceLimits, *arg0*: rti.connexdds.DataWriterResourceLimits) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.DataWriterResourceLimits) → None

Create a default DataWriterResourceLimits policy.

`__module__ = 'rti.connextdds'`

`__ne__ (self: rti.connextdds.DataWriterResourceLimits, arg0: rti.connextdds.DataWriterResourceLimits) → bool`

Test for inequality.

property autoregister_instances

Whether or not to automatically register new instances.

property cookie_max_length

Represents the maximum length in bytes of a Cookie.

property initial_active_topic_queries

Represents the initial number of active topic queries a DataWriter will manage.

property initial_batches

Represents the initial number of batches a DataWriter will manage.

property initial_concurrent_blocking_threads

The initial number of threads that are allowed to concurrently block on write call on the same DataWriter.

property initial_virtual_writers

The initial number of virtual writers supported by a DataWriter.

property initialize_writer_loaned_sample

Whether or not to initialize loaned samples returned by a DataWriter.

property instance_replacement

Sets the kinds of instances allowed to be replaced when instance resource limits are reached.

property max_active_topic_queries

Represents the maximum number of active topic queries a DataWriter will manage.

property max_app_ack_remote_readers

The maximum number of application-level acknowledging remote readers supported by a DataWriter.

property max_batches

Represents the maximum number of batches a DataWriter will manage.

property max_concurrent_blocking_threads

The maximum number of threads that are allowed to concurrently block on write call on the same DataWriter.

property max_remote_reader_filters

The maximum number of remote DataReaders for which the DataWriter will perform content-based filtering.

property max_remote_readers

The maximum number of remote readers supported by a DataWriter.

property max_virtual_writers

The maximum number of virtual writers supported by a DataWriter.

property replace_empty_instances

Whether or not to replace empty instances during instance replacement.

property writer_loaned_sample_allocation

Represents the allocation settings of loaned samples managed by a DataWriter.

```
class rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind
```

Bases: pybind11_object

```
ALIVE = <DataWriterResourceLimitsInstanceReplacementKind.ALIVE: 1>
```

```
ALIVE_OR_DISPOSED =
```

```
<DataWriterResourceLimitsInstanceReplacementKind.ALIVE_OR_DISPOSED: 5>
```

```
ALIVE_THEN_DISPOSED =
```

```
<DataWriterResourceLimitsInstanceReplacementKind.
```

```
ALIVE_THEN_DISPOSED:
```

```
3>
```

```
DISPOSED =
```

```
<DataWriterResourceLimitsInstanceReplacementKind.DISPOSED: 2>
```

```
DISPOSED_THEN_ALIVE =
```

```
<DataWriterResourceLimitsInstanceReplacementKind.
```

```
DISPOSED_THEN_ALIVE:
```

```
4>
```

```
class DataWriterResourceLimitsInstanceReplacementKind
```

Bases: pybind11_object

Members:

UNREGISTERED : Allows a DataWriter to reclaim unregistered acknowledged instances.

By default, all instance replacement kinds first attempt to reclaim an unregistered, acknowledged instance. Used in DataWriterResourceLimits.instance_replacement [default]

ALIVE : Allows a DataWriter to reclaim alive, acknowledged instances.

When an unregistered, acknowledged instance is not available to reclaim, this kind allows a DataWriter to reclaim an alive, acknowledged instance, where an alive instance is a registered, non-disposed instance. The least recently registered or written alive instance will be reclaimed.

DISPOSED : Allows a DataWriter to reclaim disposed acknowledged instances.

When an unregistered, acknowledged instance is not available to reclaim, this kind allows a DataWriter to reclaim a disposed, acknowledged instance. The least recently disposed instance will be reclaimed.

ALIVE_THEN_DISPOSED : Allows a DataWriter first to reclaim an alive, acknowledged instance, and then, if necessary, a disposed, acknowledged instance.

When an unregistered, acknowledged instance is not available to reclaim, this kind allows a DataWriter to first try reclaiming an alive, acknowledged instance. If no instance is reclaimable, then it tries reclaiming a disposed, acknowledged instance. The least recently used (i.e., registered, written, or disposed) instance will be reclaimed.

DISPOSED_THEN_ALIVE : Allows a DataWriter first to reclaim a disposed, acknowledged instance, and then, if necessary, an alive, acknowledged instance.

When an unregistered, acknowledged instance is not available to reclaim, this kind allows a DataWriter to first try reclaiming a disposed, acknowledged instance. If no instance is reclaimable, then it tries reclaiming an alive, acknowledged instance. The least recently used (i.e., disposed, registered, or written) instance will be reclaimed.

ALIVE_OR_DISPOSED : Allows a DataWriter to reclaim a either an alive acknowledged instance or a disposed acknowledged instance.

When an unregistered acknowledged instance is not available to reclaim, this kind allows a DataWriter to reclaim either an alive, acknowledged instance or a disposed, acknowledged instance. If both instance kinds are available to reclaim, the DataWriter will reclaim the least recently used (i.e. disposed, registered, or written) instance.

```
ALIVE = <DataWriterResourceLimitsInstanceReplacementKind.ALIVE:
1>
```

```
ALIVE_OR_DISPOSED =
<DataWriterResourceLimitsInstanceReplacementKind.
ALIVE_OR_DISPOSED:
5>
```

```
ALIVE_THEN_DISPOSED =
<DataWriterResourceLimitsInstanceReplacementKind.
ALIVE_THEN_DISPOSED:
3>
```

```
DISPOSED =
<DataWriterResourceLimitsInstanceReplacementKind.DISPOSED: 2>
```

```
DISPOSED_THEN_ALIVE =
<DataWriterResourceLimitsInstanceReplacementKind.
DISPOSED_THEN_ALIVE:
4>
```

```
UNREGISTERED =
<DataWriterResourceLimitsInstanceReplacementKind.UNREGISTERED:
0>
```

```
__eq__ (self: object, other: object) → bool
```

```

__getstate__(self: object) → int

__hash__(self: object) → int

__index__(self: rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind) → int

__init__(self: rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind, value: int) → None

__int__(self: rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind) → int

__members__ = {'ALIVE':
<DataWriterResourceLimitsInstanceReplacementKind.ALIVE: 1>,
'ALIVE_OR_DISPOSED':
<DataWriterResourceLimitsInstanceReplacementKind.ALIVE_OR_DISPOSED: 5>, 'ALIVE_THEN_DISPOSED':
<DataWriterResourceLimitsInstanceReplacementKind.ALIVE_THEN_DISPOSED: 3>, 'DISPOSED':
<DataWriterResourceLimitsInstanceReplacementKind.DISPOSED: 2>,
'DISPOSED_THEN_ALIVE':
<DataWriterResourceLimitsInstanceReplacementKind.DISPOSED_THEN_ALIVE: 4>, 'UNREGISTERED':
<DataWriterResourceLimitsInstanceReplacementKind.UNREGISTERED: 0>}

__module__ = 'rti.connextdds'

__ne__(self: object, other: object) → bool

__repr__(self: object) → str

__setstate__(self: rti.connextdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind, state: int) → None

__str__()
    name(self: handle) -> str

property name

property value

UNREGISTERED =
<DataWriterResourceLimitsInstanceReplacementKind.UNREGISTERED: 0>

```

__eq__ (*self*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind, *arg0*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind) → bool

Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind, *arg0*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind) → bool

Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind, *arg0*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind, *arg0*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind) → *rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind*

Int conversion.

__le__ (*self*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind, *arg0*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind, *arg0*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind, *arg0*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class `rti.connexdds.DataWriterSeq`Bases: `pybind11_object``__add__` (*self*: `List[rti.connexdds.DataWriter]`, *arg0*: `List[rti.connexdds.DataWriter]`) → `List[rti.connexdds.DataWriter]``__bool__` (*self*: `List[rti.connexdds.DataWriter]`) → `bool`
Check whether the list is nonempty`__contains__` (*self*: `List[rti.connexdds.DataWriter]`, *x*: `rti.connexdds.DataWriter`) → `bool`
Return true the container contains *x*`__delitem__` (**args*, ***kwargs*)
Overloaded function.1. `__delitem__(self: List[rti.connexdds.DataWriter], arg0: int) -> None`Delete the list elements at index *i*2. `__delitem__(self: List[rti.connexdds.DataWriter], arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (*self*: `List[rti.connexdds.DataWriter]`, *arg0*: `List[rti.connexdds.DataWriter]`) → `bool``__getitem__` (**args*, ***kwargs*)
Overloaded function.1. `__getitem__(self: List[rti.connexdds.DataWriter], s: slice) -> List[rti.connexdds.DataWriter]`

Retrieve list elements using a slice object

2. `__getitem__(self: List[rti.connexdds.DataWriter], arg0: int) -> rti.connexdds.DataWriter``__hash__` = `None``__iadd__` (*self*: `List[rti.connexdds.DataWriter]`, *arg0*: `List[rti.connexdds.DataWriter]`) → `List[rti.connexdds.DataWriter]``__imul__` (*self*: `List[rti.connexdds.DataWriter]`, *arg0*: `int`) → `List[rti.connexdds.DataWriter]``__init__` (**args*, ***kwargs*)
Overloaded function.1. `__init__(self: rti.connexdds.DataWriterSeq) -> None`2. `__init__(self: rti.connexdds.DataWriterSeq, arg0: List[rti.connexdds.DataWriter]) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.DataWriterSeq, arg0: Iterable) -> None``__iter__` (*self*: `List[rti.connexdds.DataWriter]`) → `Iterator``__len__` (*self*: `List[rti.connexdds.DataWriter]`) → `int`

__module__ = 'rti.connexdds'

__mul__ (*self*: List[rti.connexdds.DataWriter], *arg0*: int) → List[rti.connexdds.DataWriter]

__ne__ (*self*: List[rti.connexdds.DataWriter], *arg0*: List[rti.connexdds.DataWriter]) → bool

__rmul__ (*self*: List[rti.connexdds.DataWriter], *arg0*: int) → List[rti.connexdds.DataWriter]

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: List[rti.connexdds.DataWriter], *arg0*: int, *arg1*: rti.connexdds.DataWriter) -> None
2. **__setitem__**(*self*: List[rti.connexdds.DataWriter], *arg0*: slice, *arg1*: List[rti.connexdds.DataWriter]) -> None

Assign list elements using a slice object

append (*self*: List[rti.connexdds.DataWriter], *x*: rti.connexdds.DataWriter) → None

Add an item to the end of the list

clear (*self*: List[rti.connexdds.DataWriter]) → None

Clear the contents

count (*self*: List[rti.connexdds.DataWriter], *x*: rti.connexdds.DataWriter) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: List[rti.connexdds.DataWriter], *L*: List[rti.connexdds.DataWriter]) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: List[rti.connexdds.DataWriter], *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: List[rti.connexdds.DataWriter], *i*: int, *x*: rti.connexdds.DataWriter) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: List[rti.connexdds.DataWriter]) -> rti.connexdds.DataWriter

Remove and return the last item

2. **pop**(*self*: List[rti.connexdds.DataWriter], *i*: int) -> rti.connexdds.DataWriter

Remove and return the item at index *i*

remove (*self*: List[rti.connexdds.DataWriter], *x*: rti.connexdds.DataWriter) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

```

class rti.connexdds.DataWriterShmemRefTransferModeSettings
    Bases: pybind11_object

    __eq__ (self: rti.connexdds.DataWriterShmemRefTransferModeSettings, arg0:
            rti.connexdds.DataWriterShmemRefTransferModeSettings) → bool
        Test for equality.

    __hash__ = None

    __init__ (self: rti.connexdds.DataWriterShmemRefTransferModeSettings) → None
        Creates an instance with the default settings.

    __module__ = 'rti.connexdds'

    __ne__ (self: rti.connexdds.DataWriterShmemRefTransferModeSettings, arg0:
            rti.connexdds.DataWriterShmemRefTransferModeSettings) → bool
        Test for inequality.

    property enable_data_consistency_check
        Controls if samples can be checked for consistency.

class rti.connexdds.DataWriterTransferMode
    Bases: pybind11_object

    __eq__ (self: rti.connexdds.DataWriterTransferMode, arg0:
            rti.connexdds.DataWriterTransferMode) → bool
        Test for equality.

    __hash__ = None

    __init__ (self: rti.connexdds.DataWriterTransferMode) → None
        Creates a DataWriterTransferMode qos policy with default values.

    __module__ = 'rti.connexdds'

    __ne__ (self: rti.connexdds.DataWriterTransferMode, arg0:
            rti.connexdds.DataWriterTransferMode) → bool
        Test for inequality.

    property shmem_ref_settings
        Settings related to transferring data using shared memory references.

class rti.connexdds.Database
    Bases: pybind11_object

    __eq__ (self: rti.connexdds.Database, arg0: rti.connexdds.Database) → bool
        Test for equality.

    __hash__ = None

    __init__ (self: rti.connexdds.Database) → None
        Create a Database QoS policy with default settings.

```

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.Database, arg0: rti.connexdds.Database) → bool  
Test for inequality.
```

```
property cleanup_period  
Get/set the cleanup period.
```

```
property initial_records  
Get/set the number of database records to be allocated initially.
```

```
property initial_weak_references  
Get/set the initial number of weak references.
```

```
property max_skiplist_level  
Get/set the skiplist level.
```

```
property max_weak_references  
Get/set the maximum number of weak references.
```

```
property shutdown_cleanup_period  
Get/set the shutdown cleanup period.
```

```
property shutdown_timeout  
Get/set the shutdown timeout.
```

```
property thread  
Get/set the thread settings for the Database thread.
```

```
class rti.connexdds.Deadline
```

```
Bases: pybind11_object
```

```
__eq__ (self: rti.connexdds.Deadline, arg0: rti.connexdds.Deadline) → bool  
Test for equality.
```

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

```
Overloaded function.
```

```
1. __init__(self: rti.connexdds.Deadline) -> None
```

```
Creates the default deadline, with an infinite period.
```

```
2. __init__(self: rti.connexdds.Deadline, period: rti.connexdds.Duration) -> None
```

```
Creates a deadline policy with the specified period.
```

```
3. __init__(self: rti.connexdds.Deadline, sec: int, nanosec: int) -> None
```

```
Creates a deadline policy with the specified period.
```

```
__module__ = 'rti.connexdds'
```


`__ne__` (*self*: rti.connexdds.Deadline, *arg0*: rti.connexdds.Deadline) → bool
 Test for inequality.

property period

The duration of the deadline period.

class rti.connexdds.DestinationOrder

Bases: pybind11_object

`__eq__` (*self*: rti.connexdds.DestinationOrder, *arg0*: rti.connexdds.DestinationOrder) → bool
 Test for equality.

`__hash__` = None

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.DestinationOrder) -> None`

Creates the default policy.

2. `__init__(self: rti.connexdds.DestinationOrder, kind: rti.connexdds.DestinationOrderKind) -> None`

Creates a policy with the specified destination order kind.

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.DestinationOrder, *arg0*: rti.connexdds.DestinationOrder) → bool
 Test for inequality.

property kind

The destination order kind.

property scope

The destination order scope.

property source_timestamp_tolerance

The allowed tolerance between source timestamps of consecutive samples.

class rti.connexdds.DestinationOrderKind

Bases: pybind11_object

`BY_RECEPTION_TIMESTAMP` =

<DestinationOrderKind.BY_RECEPTION_TIMESTAMP: 0>

`BY_SOURCE_TIMESTAMP` = <DestinationOrderKind.BY_SOURCE_TIMESTAMP: 1>

class DestinationOrderKind

Bases: pybind11_object

Members:

`BY_RECEPTION_TIMESTAMP` : [default] Indicates that data is ordered based on the reception time at each Subscriber.

Since each subscriber may receive the data at different times there is no guarantee that the changes will be seen in the same order. Consequently, it is possible for each subscriber to end up with a different final value for the data.

BY_SOURCE_TIMESTAMP : Indicates that data is ordered based on a time-stamp placed at the source (by RTI Connex or by the application).

In any case this guarantees a consistent final value for the data in all subscribers.

Note: If Batching is needed along with `DestinationOrderKind.BY_SOURCE_TIMESTAMP` and `DestinationOrderScopeKind.INSTANCE`, then the `Batch.source_timestamp_resolution` and `Batch.thread_safe_write` setting of `Batch` should be set to `Duration.zero()` and `true` respectively.

```

BY_RECEPTION_TIMESTAMP =
<DestinationOrderKind.BY_RECEPTION_TIMESTAMP: 0>

BY_SOURCE_TIMESTAMP =
<DestinationOrderKind.BY_SOURCE_TIMESTAMP: 1>

__eq__(self: object, other: object) → bool

__getstate__(self: object) → int

__hash__(self: object) → int

__index__(self: rti.connextdds.DestinationOrderKind.DestinationOrderKind) → int

__init__(self: rti.connextdds.DestinationOrderKind.DestinationOrderKind, value: int) →
    None

__int__(self: rti.connextdds.DestinationOrderKind.DestinationOrderKind) → int

__members__ = {'BY_RECEPTION_TIMESTAMP':
<DestinationOrderKind.BY_RECEPTION_TIMESTAMP: 0>,
'BY_SOURCE_TIMESTAMP':
<DestinationOrderKind.BY_SOURCE_TIMESTAMP: 1>}

__module__ = 'rti.connextdds'

__ne__(self: object, other: object) → bool

__repr__(self: object) → str

__setstate__(self: rti.connextdds.DestinationOrderKind.DestinationOrderKind, state: int)
    → None

__str__()
    name(self: handle) -> str

property name

property value

```

__eq__ (*self*: rti.connexdds.DestinationOrderKind, *arg0*: rti.connexdds.DestinationOrderKind) → bool

Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.DestinationOrderKind, *arg0*: rti.connexdds.DestinationOrderKind) → bool

Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.DestinationOrderKind, *arg0*: rti.connexdds.DestinationOrderKind) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DestinationOrderKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.DestinationOrderKind, *arg0*: rti.connexdds.DestinationOrderKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.DestinationOrderKind) → *rti.connexdds.DestinationOrderKind.DestinationOrderKind*

Int conversion.

__le__ (*self*: rti.connexdds.DestinationOrderKind, *arg0*: rti.connexdds.DestinationOrderKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.DestinationOrderKind, *arg0*: rti.connexdds.DestinationOrderKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DestinationOrderKind, *arg0*: rti.connexdds.DestinationOrderKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.DestinationOrderKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.DestinationOrderScopeKind

Bases: pybind11_object

class DestinationOrderScopeKind

Bases: pybind11_object

Members:

INSTANCE : [default] Indicates that data is ordered on a per instance basis if used along with DestinationOrderKind.BY_SOURCE_TIMESTAMP.

The source timestamp of the current sample is compared to the source timestamp of the previously received sample for the same instance. The tolerance check is also applied per instance.

TOPIC : Indicates that data is ordered on a per topic basis if used along with DestinationOrderKind.BY_SOURCE_TIMESTAMP.

The source timestamp of the current sample is compared to the source timestamp of the previously received sample for the same topic. The tolerance check is also applied per topic.

INSTANCE = <DestinationOrderScopeKind.INSTANCE: 0>

TOPIC = <DestinationOrderScopeKind.TOPIC: 1>

__eq__ (*self: object, other: object*) → bool

__getstate__ (*self: object*) → int

__hash__ (*self: object*) → int

__index__ (*self: rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind*) → int

__init__ (*self: rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind, value: int*) → None

__int__ (*self: rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind*) → int

__members__ = {'INSTANCE': <DestinationOrderScopeKind.INSTANCE: 0>, 'TOPIC': <DestinationOrderScopeKind.TOPIC: 1>}

__module__ = 'rti.connextdds'

__ne__ (*self: object, other: object*) → bool

__repr__ (*self: object*) → str

__setstate__ (*self: rti.connextdds.DestinationOrderScopeKind.DestinationOrderScopeKind, state: int*) → None

__str__ ()
name(*self: handle*) -> str

property name

property value

INSTANCE = <DestinationOrderScopeKind.INSTANCE: 0>

TOPIC = <DestinationOrderScopeKind.TOPIC: 1>

__eq__ (*self*: rti.connexdds.DestinationOrderScopeKind, *arg0*: rti.connexdds.DestinationOrderScopeKind) → bool

Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.DestinationOrderScopeKind, *arg0*: rti.connexdds.DestinationOrderScopeKind) → bool

Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.DestinationOrderScopeKind, *arg0*: rti.connexdds.DestinationOrderScopeKind) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DestinationOrderScopeKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.DestinationOrderScopeKind, *arg0*: rti.connexdds.DestinationOrderScopeKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.DestinationOrderScopeKind) → *rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind*

Int conversion.

__le__ (*self*: rti.connexdds.DestinationOrderScopeKind, *arg0*: rti.connexdds.DestinationOrderScopeKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.DestinationOrderScopeKind, *arg0*: rti.connexdds.DestinationOrderScopeKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DestinationOrderScopeKind, *arg0*: rti.connexdds.DestinationOrderScopeKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.DestinationOrderScopeKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class `rti.connexdds.Discovery`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.Discovery`, *arg0*: `rti.connexdds.Discovery`) → bool

Test for equality.

__hash__ = None

__init__ (*self*: `rti.connexdds.Discovery`) → None

Creates the default policy.

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.Discovery`, *arg0*: `rti.connexdds.Discovery`) → bool

Test for inequality.

property accept_unknown_peers

Whether to accept a new participant that is not in the initial peers list.

property enable_endpoint_discovery

Whether to automatically enable endpoint discovery for all the remote participants.

property enabled_transports

The transports (by their aliases) available for the discovery mechanism.

This property's getter returns a deep copy.

property initial_peers

The initial list of peers that the discovery mechanism will contact to announce this DomainParticipant.

This property's getter returns a deep copy.

property metatraffic_transport_priority

The transport priority to use for the Discovery meta-traffic.

property multicast_receive_addresses

The multicast group addresses on which discovery-related meta-traffic can be received by the DomainParticipant.

This property's getter returns a deep copy.

class `rti.connexdds.DiscoveryConfig`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.DiscoveryConfig`, *arg0*: `rti.connexdds.DiscoveryConfig`) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.DiscoveryConfig) → None

Creates the default policy.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DiscoveryConfig, *arg0*: rti.connexdds.DiscoveryConfig) → bool

Test for inequality.

property asynchronous_publisher

Asynchronous publishing settings for the discovery Publisher and all entities that are created by it.

property builtin_discovery_plugins

The kind mask for built-in discovery plugins.

property default_domain_announcement_period

The period to announce a participant to the default domain 0.

This property's getter returns a deep copy.

property dns_tracker_polling_period

Duration that specifies the period used by the DNS tracker to poll the DNS service and check for changes in the hostnames.

property enabled_builtin_channels

The mask specifying which built-in channels should be enabled.

property endpoint_type_object_lb_serialization_threshold

Option to reduce the size required to propagate a TypeObject in Simple Endpoint Discovery.

property ignore_default_domain_announcements

Used to ignore the announcements received by a participant on the default domain 0 corresponding to participants running on domains IDs other than 0.

property initial_participant_announcements

The number of initial announcements sent when a participant is first enabled or when a remote participant is newly discovered.

property locator_reachability_assert_period

Period at which this DomainParticipant will assert the locators discovered from other DomainParticipants.

This property's getter returns a deep copy.

property locator_reachability_change_detection_period

Period at which this DomainParticipant will check if its locators are reachable from other DomainParticipants.

This property's getter returns a deep copy.

property locator_reachability_lease_duration

The time period after which other DomainParticipants can consider one of their locators as “un-reachable” if they do not receive a REACHABILITY PING from this DomainParticipant.

This property's getter returns a deep copy.

property max_initial_participant_announcement_period

The maximum period between initial announcements when a participant is first enabled or when a remote participant is newly discovered.

property max_liveliness_loss_detection_period

The period to assert liveliness for the participant.

This property's getter returns a deep copy.

property min_initial_participant_announcement_period

The minimum period between initial announcements when a participant is first enabled or when a remote participant is newly discovered.

This property's getter returns a deep copy.

property new_remote_participant_announcements

The number of participant announcements sent when a remote participant is newly discovered.

property participant_announcement_period

The period at which a participant announces itself to potential peers when using the Simple Participant Discovery Protocol 2.0 (SPDP2)

property participant_liveliness_assert_period

The period to assert liveliness for the participant.

This property's getter returns a deep copy.

property participant_liveliness_lease_duration

The liveliness lease duration for the participant.

property participant_message_reader

Reliability policy for a built-in participant message reader.

property participant_message_reader_reliability_kind

Reliability policy for a built-in participant message reader.

property participant_message_writer

Reliability policy for a built-in participant message writer.

property participant_reader_resource_limits

Resource limits.

property publication_reader

RTPS protocol-related configuration settings for the RTPS reliable reader associated to a built-in publication reader.

property publication_reader_resource_limits

Publication reader resource limits.

property publication_writer

RTPS protocol-related configuration settings for the RTPS reliable writer associated to a built-in publication reader.

property publication_writer_data_lifecycle

riter data lifecycle settings for a built-in publication writer.

property publication_writer_publish_mode

Publish mode policy for the built-in publication writer.

property remote_participant_purge_kind

The participant's behavior for maintaining knowledge of remote participants (and their contained entities) with which discovery communication has been lost.

property secure_volatile_reader

RTPS reliable reader protocol-related configuration settings for the built-in secure volatile reader.

property secure_volatile_writer

RTPS protocol-related configuration settings for the RTPS reliable writer associated with the built-in secure volatile writer.

property secure_volatile_writer_publish_mode

Publish mode policy for the built-in secure volatile writer.

property service_request_reader

RTPS protocol-related configuration settings for the RTPS reliable reader associated to a built-in publication reader.

property service_request_writer

RTPS protocol-related configuration settings for the RTPS reliable writer associated to the built-in ServiceRequest writer.

property service_request_writer_data_lifecycle

Writer data lifecycle settings for a built-in ServiceRequest writer.

property service_request_writer_publish_mode

Publish mode policy for the built-in service request writer.

property subscription_reader

RTPS protocol-related configuration settings for the RTPS reliable reader associated to a built-in subscription reader.

property subscription_reader_resource_limits

Subscription reader resource limits.

property subscription_writer

RTPS protocol-related configuration settings for the RTPS reliable writer associated to a built-in subscription reader.

property subscription_writer_data_lifecycle

riter data lifecycle settings for a built-in subscription writer.

property subscription_writer_publish_mode

Publish mode policy for the built-in subscription writer.

```
class rti.connexdds.DiscoveryConfigBuiltinChannelKindMask
```

```
Bases: pybind11_object
```

```
ALL = 1111111
```

```
NONE = 0000000
```

```
SERVICE_REQUEST = 0000011
```

```
__and__ (self: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, arg0:
         rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) →
         rti.connexdds.DiscoveryConfigBuiltinChannelKindMask
```

Bitwise logical AND of masks.

```
__bool__ (self: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → bool
```

Test if any bits are set.

```
__contains__ (self: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, arg0:
              rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → bool
```

```
__eq__ (self: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, arg0:
         rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → bool
```

Compare masks for equality.

```
__getitem__ (self: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, arg0: int) → bool
```

Get individual mask bit.

```
__hash__ = None
```

```
__iand__ (self: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, arg0:
          rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) →
          rti.connexdds.DiscoveryConfigBuiltinChannelKindMask
```

Set mask to logical AND with another mask.

```
__ilshift__ (self: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, arg0: int) →
              rti.connexdds.DiscoveryConfigBuiltinChannelKindMask
```

Left shift bits in mask.

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) -> None`

Create a `DiscoveryConfigBuiltinChannelKindMask` equivalent to `DiscoveryConfigBuiltinChannelKindMask.NONE`

2. `__init__(self: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, value: int) -> None`

Creates a mask from the bits in an integer.

```
__int__ (self: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → int
```

Convert mask to int.

__ior__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, *arg0*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → *rti.connexdds.DiscoveryConfigBuiltinChannelKindMask*

Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, *arg0*: int) → *rti.connexdds.DiscoveryConfigBuiltinChannelKindMask*

Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, *arg0*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → *rti.connexdds.DiscoveryConfigBuiltinChannelKindMask*

Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, *arg0*: int) → *rti.connexdds.DiscoveryConfigBuiltinChannelKindMask*

Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, *arg0*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → bool

Compare masks for inequality.

__or__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, *arg0*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → *rti.connexdds.DiscoveryConfigBuiltinChannelKindMask*

Bitwise logical OR of masks.

__repr__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → str

Convert mask to string.

__rshift__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, *arg0*: int) → *rti.connexdds.DiscoveryConfigBuiltinChannelKindMask*

Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, *arg0*: int, *arg1*: bool) → None

Set individual mask bit

__str__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → str

Convert mask to string.

__xor__ (*self*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask, *arg0*: rti.connexdds.DiscoveryConfigBuiltinChannelKindMask) → *rti.connexdds.DiscoveryConfigBuiltinChannelKindMask*

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (*args, **kwargs)

Overloaded function.

1. flip(self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) -> rti.connextdds.DiscoveryConfigBuiltinChannelKindMask

Flip all bits in the mask.

2. flip(self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, pos: int) -> rti.connextdds.DiscoveryConfigBuiltinChannelKindMask

Flip the mask bit at the specified position.

reset (*args, **kwargs)

Overloaded function.

1. reset(self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) -> rti.connextdds.DiscoveryConfigBuiltinChannelKindMask

Clear all bits in the mask.

2. reset(self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, pos: int) -> rti.connextdds.DiscoveryConfigBuiltinChannelKindMask

Clear the mask bit at the specified position.

set (*args, **kwargs)

Overloaded function.

1. set(self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) -> rti.connextdds.DiscoveryConfigBuiltinChannelKindMask

Set all bits in the mask.

2. set(self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, pos: int, value: bool = True) -> rti.connextdds.DiscoveryConfigBuiltinChannelKindMask

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask, pos: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) → bool

Test if all bits are set.

test_any (self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) → bool

Test if any bits are set.

test_none (self: rti.connextdds.DiscoveryConfigBuiltinChannelKindMask) → bool

Test if none of the bits are set.

```
class rti.connextdds.DiscoveryConfigBuiltinPluginKindMask
```

```
Bases: pybind11_object
```

```
NONE = 00000
```

```
SDP = 00011
```

```
SDP2 = 10010
```

```
SEDP = 00010
```

```
SPDP = 00001
```

```
SPDP2 = 10000
```

```
__and__ (self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, arg0:  
          rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) →  
          rti.connextdds.DiscoveryConfigBuiltinPluginKindMask
```

Bitwise logical AND of masks.

```
__bool__ (self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → bool  
Test if any bits are set.
```

```
__contains__ (self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, arg0:  
              rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → bool
```

```
__eq__ (self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, arg0:  
        rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) → bool
```

Compare masks for equality.

```
__getitem__ (self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, arg0: int) → bool  
Get individual mask bit.
```

```
__hash__ = None
```

```
__iand__ (self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, arg0:  
          rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) →  
          rti.connextdds.DiscoveryConfigBuiltinPluginKindMask
```

Set mask to logical AND with another mask.

```
__ilshift__ (self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, arg0: int) →  
              rti.connextdds.DiscoveryConfigBuiltinPluginKindMask
```

Left shift bits in mask.

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask) -> None`

Create a `DiscoveryConfigBuiltinPluginKindMask` equivalent to `DiscoveryConfigBuiltinPluginKindMask.NONE`

2. `__init__(self: rti.connextdds.DiscoveryConfigBuiltinPluginKindMask, value: int) -> None`

Creates a mask from the bits in an integer.

__int__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → int
Convert mask to int.

__ior__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, *arg0*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, *arg0*: int) → rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, *arg0*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, *arg0*: int) → rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, *arg0*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → bool

Compare masks for inequality.

__or__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, *arg0*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Bitwise logical OR of masks.

__repr__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → str
Convert mask to string.

__rshift__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, *arg0*: int) → rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, *arg0*: int, *arg1*: bool) → None

Set individual mask bit

__str__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → str
Convert mask to string.

__xor__ (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, *arg0*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (*args, **kwargs)

Overloaded function.

1. flip(self: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) -> rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Flip all bits in the mask.

2. flip(self: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, pos: int) -> rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Flip the mask bit at the specified position.

reset (*args, **kwargs)

Overloaded function.

1. reset(self: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) -> rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Clear all bits in the mask.

2. reset(self: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, pos: int) -> rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Clear the mask bit at the specified position.

set (*args, **kwargs)

Overloaded function.

1. set(self: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) -> rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Set all bits in the mask.

2. set(self: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, pos: int, value: bool = True) -> rti.connexdds.DiscoveryConfigBuiltinPluginKindMask

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (self: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask, pos: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (self: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → bool

Test if all bits are set.

test_any (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → bool
 Test if any bits are set.

test_none (*self*: rti.connexdds.DiscoveryConfigBuiltinPluginKindMask) → bool
 Test if none of the bits are set.

class rti.connexdds.DomainParticipant

Bases: *IEntity*

Container for all Entity objects.

- It acts as a container for all other Entity objects.
- It acts as a factory for the Publisher, Subscriber, Topic and Entity objects.
- It represents the participation of the application on a communication plane that isolates applications running on the same set of physical computers from each other. A domain establishes a virtual network linking all applications that share the same domainId and isolating them from applications running on different domains. In this way, several independent distributed applications can coexist in the same physical network without interfering, or even being aware of each other.
- It provides administration services in the domain, offering operations that allow the application to ignore locally any information about a given participant (DomainParticipant.ignore), publication (ignore_publication), subscription (ignore_subscription) or topic (ignore_topic).

__enter__ (*self*: rti.connexdds.DomainParticipant) → *rti.connexdds.DomainParticipant*
 Enter a context for this Domain Participant, to be cleaned up on exiting context

__eq__ (*self*: rti.connexdds.DomainParticipant, *arg0*: rti.connexdds.DomainParticipant) → bool
 Test for equality.

__exit__ (*self*: rti.connexdds.DomainParticipant, *arg0*: object, *arg1*: object, *arg2*: object) → None
 Exit the context for this Domain Participant, cleaning up resources.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DomainParticipant, domain_id: int) -> None

Create a new DomainParticipant with default QoS.

2. **__init__**(*self*: rti.connexdds.DomainParticipant, domain_id: int, qos: rti.connexdds.DomainParticipantQos, listener: rti.connexdds.DomainParticipantListener = None, mask: rti.connexdds.StatusMask = StatusMask.ALL) -> None

Create a new DomainParticipant

3. **__init__**(*self*: rti.connexdds.DomainParticipant, entity: rti.connexdds.IEntity) -> None

Downcast an IEntity to a DomainParticipant.

__lshift__ (*self*: rti.connextdds.DomainParticipant, *arg0*: rti.connextdds.DomainParticipantQos)
 → rti.connextdds.DomainParticipant

Set the domain participant's QoS.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.DomainParticipant, *arg0*: rti.connextdds.DomainParticipant) → bool

Test for inequality.

__rshift__ (*self*: rti.connextdds.DomainParticipant, *arg0*: rti.connextdds.DomainParticipantQos)
 → rti.connextdds.DomainParticipant

Get the domain participant's QoS.

add_peer (*self*: rti.connextdds.DomainParticipant, *peer*: str) → None

Attempt to contact an additional peer participant.

add_peers (*self*: rti.connextdds.DomainParticipant, *peers*: rti.connextdds.StringSeq) → None

Add a sequence of peers to be contacted.

assert_liveliness (*self*: rti.connextdds.DomainParticipant) → None

Manually assert the liveliness of the DomainParticipant.

banish_ignored_participants (*self*: rti.connextdds.DomainParticipant) → None

Prevents ignored remote DomainParticipants from receiving traffic from the local DomainParticipant.

property builtin_subscriber

Get the built-in subscriber for the DomainParticipant.

close_contained_entities (*self*: rti.connextdds.DomainParticipant) → None

Destroy all contained entities.

contains_entity (*self*: rti.connextdds.DomainParticipant, *handle*:
 rti.connextdds.InstanceHandle) → bool

Check whether or not the given handle represents an Entity that was created from the DomainParticipant.

property current_time

Get the current time.

property default_datareader_qos

The default DataReaderQos.

This property's getter returns a deep copy.

property default_datawriter_qos

The default DataWriterQos.

This property's getter returns a deep copy.

default_participant_qos = <rti.connextdds.DomainParticipantQos
 object>

property default_publisher_qos

The default PublisherQos.

This property's getter returns a deep copy.

property default_subscriber_qos

The default SubscriberQos.

This property's getter returns a deep copy.

property default_topic_qos

The default TopicQos.

This property's getter returns a deep copy.

delete_durable_subscription (*self*: rti.connextdds.DomainParticipant, *group*: rti.connextdds.EndpointGroup) → None

Deletes an existing Durable Subscription on all Persistence Services.

discovered_participant_data (**args*, ***kwargs*)

Overloaded function.

1. `discovered_participant_data(self: rti.connextdds.DomainParticipant, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.ParticipantBuiltinTopicData`

Retrieve DomainParticipant information by handle.

2. `discovered_participant_data(self: rti.connextdds.DomainParticipant, handles: rti.connextdds.InstanceHandleSeq) -> rti.connextdds.ParticipantBuiltinTopicDataSeq`

Retrieve DomainParticipant information with a sequence of handles.

discovered_participant_subject_name (*self*: rti.connextdds.DomainParticipant, *handle*: rti.connextdds.InstanceHandle) → Optional[str]

Returns the entity name for the specified DomainParticipant InstanceHandle.

discovered_participants (*self*: rti.connextdds.DomainParticipant) → rti.connextdds.InstanceHandleSeq

Retrieves the instance handles of other DomainParticipants discovered by this one.

discovered_participants_from_subject_name (*self*: rti.connextdds.DomainParticipant, *subject_name*: Optional[str]) → rti.connextdds.InstanceHandleSeq

Returns the list of InstanceHandles corresponding to participants with the given entity name.

discovered_topic_data (**args*, ***kwargs*)

Overloaded function.

1. `discovered_topic_data(self: rti.connextdds.DomainParticipant, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.TopicBuiltinTopicData`

Get information about a discovered topic using its handle.

2. `discovered_topic_data(self: rti.connextdds.DomainParticipant, topic: rti.connextdds.IEntity) -> rti.connextdds.TopicBuiltinTopicData`

Get information about a discovered topic.

3. `discovered_topic_data(self: rti.connextdds.DomainParticipant, handles: rti.connextdds.InstanceHandleSeq) -> rti.connextdds.TopicBuiltinTopicDataSeq`

Get information about a discovered topics with their handles.

4. `discovered_topic_data(self: rti.connextdds.DomainParticipant) -> rti.connextdds.TopicBuiltinTopicDataSeq`

Get information about all discovered topics.

discovered_topics (*self*: rti.connextdds.DomainParticipant) →
rti.connextdds.InstanceHandleSeq

Get all Topic handles discovered by this DomainParticipant.

property domain_id

The unique domain identifier.

static finalize_participant_factory () → None

Finalize the DomainParticipantFactory

static find (*args, **kwargs)

Overloaded function.

1. `find(name: str) -> Optional[rti.connextdds.DomainParticipant]`

Find a local DomainParticipant by its name.

2. `find() -> rti.connextdds.DomainParticipantSeq`

Find all local DomainParticipants.

3. `find(domain_id: int) -> Optional[rti.connextdds.DomainParticipant]`

Find a local DomainParticipant with the given domain ID. If more than one DomainParticipant on the same domain exists in the application, it is not specified which will be returned.

find_contentfilter (*self*: rti.connextdds.DomainParticipant, *name*: str) →
Optional[rti.connextdds.ContentFilterBase]

Find content filter previously registered to this DomainParticipant.

find_datareader (*self*: rti.connextdds.DomainParticipant, *name*: str) →
Optional[rti.connextdds.AnyDataReader]

Find a DataReader by its name.

find_datawriter (*self*: rti.connextdds.DomainParticipant, *name*: str) →
Optional[rti.connextdds.AnyDataWriter]

Find a DataWriter by its name.

find_flow_controller (*self*: rti.connexdds.DomainParticipant, *name*: str) →
Optional[rti.connexdds.FlowController]

Find a FlowController configured in this DomainParticipant.

find_publisher (*self*: rti.connexdds.DomainParticipant, *name*: str) →
Optional[rti.connexdds.Publisher]

Lookup a Publisher within the DomainParticipant by its entity name.

find_publishers (*self*: rti.connexdds.DomainParticipant) → rti.connexdds.PublisherSeq

Find all Publishers within the DomainParticipant.

find_registered_content_filters (*self*: rti.connexdds.DomainParticipant) →
rti.connexdds.StringSeq

Retrieve a list of all registered content filter names.

find_subscriber (*self*: rti.connexdds.DomainParticipant, *name*: str) →
Optional[rti.connexdds.Subscriber]

Find a Subscriber in the DomainParticipant by its entity name.

find_subscribers (*self*: rti.connexdds.DomainParticipant) → rti.connexdds.SubscriberSeq

Find all subscribers within the DomainParticipant.

find_topics (*self*: rti.connexdds.DomainParticipant) → rti.connexdds.AnyTopicSeq

Find all Topics in the DomainParticipant.

ignore_datareader (*self*: rti.connexdds.DomainParticipant, *handle*:
rti.connexdds.InstanceHandle) → None

Ignore a DataReader matching the provided handle.

ignore_datareaders (*self*: rti.connexdds.DomainParticipant, *handles*:
rti.connexdds.InstanceHandleSeq) → None

Ignore a list of DataReaders specified by their handles.

ignore_datawriter (*self*: rti.connexdds.DomainParticipant, *handle*:
rti.connexdds.InstanceHandle) → None

Ignore a DataWriter matching the provided handle.

ignore_datawriters (*self*: rti.connexdds.DomainParticipant, *handles*:
rti.connexdds.InstanceHandleSeq) → None

Ignore a list of DataWriters specified by their handles.

ignore_participant (*self*: rti.connexdds.DomainParticipant, *handle*:
rti.connexdds.InstanceHandle) → None

Ignore a DomainParticipant given it's handle.

ignore_participants (*self*: rti.connexdds.DomainParticipant, *arg0*:
rti.connexdds.InstanceHandleSeq) → None

Ignore DomainParticipants given a list of handles.

ignore_topic (*self*: rti.connextdds.DomainParticipant, *handle*: rti.connextdds.InstanceHandle) → None

Ignore a Topic matching the provided handle.

ignore_topics (*self*: rti.connextdds.DomainParticipant, *handles*: rti.connextdds.InstanceHandleSeq) → None

Ignore a list of Topics specified by their handles.

property implicit_publisher

Get the implicit Publisher for the DomainParticipant.

property implicit_subscriber

Get the implicit Subscriber for the DomainParticipant.

is_type_registered (*self*: rti.connextdds.DomainParticipant, *name*: str) → bool

Check if a type has been registered to this DomainParticipant.

property listener

Get the listener.

participant_factory_qos =

<rti.connextdds.DomainParticipantFactoryQos object>

property participant_protocol_status

Get the protocol status for this participant

property participant_reader

Get the DomainParticipant built-in topic reader.

property publication_reader

Get the publication built-in topic reader.

property qos

Get the domain participant's QoS.

This property's getter returns a deep copy.

register_contentfilter (*self*: rti.connextdds.DomainParticipant, *filter*: rti.connextdds.ContentFilterBase, *name*: str) → None

Register a content filter which can be used to create a ContentFilteredTopic.

register_durable_subscription (*self*: rti.connextdds.DomainParticipant, *group*: rti.connextdds.EndpointGroup, *topic_name*: str) → None

Registers a Durable Subscription on the specified Topic on all Persistence Services

static register_idl_type (*type*: object, *registered_type_name*: str) → None

Registers a python class so it can be used in XML-based applications and referred to by its registered name.

register_type (*self*: rti.connexdds.DomainParticipant, *name*: str, *type*: rti.connexdds.DynamicType, *serialization_property*: rti.connexdds.DynamicDataTypeSerializationProperty = *DynamicDataTypeSerializationProperty.DEFAULT*) → None

Registers a DynamicType with specific serialization properties.

remove_peer (*self*: rti.connexdds.DomainParticipant, *peer*: str) → None

Remove a peer participant from this list that this DomainParticipant will attempt to communicate with.

remove_peers (*self*: rti.connexdds.DomainParticipant, *peers*: rti.connexdds.StringSeq) → None

Remove a sequence of peers from the contact list.

resume_endpoint_discovery (*self*: rti.connexdds.DomainParticipant, *handle*: rti.connexdds.InstanceHandle) → None

Initiates endpoint discovery with the remote DomainParticipant identified by its InstanceHandle.

property_service_request_reader

Get the ServiceRequest built-in topic reader.

set_listener (*self*: rti.connexdds.DomainParticipant, *listener*: rti.connexdds.DomainParticipantListener, *event_mask*: rti.connexdds.StatusMask) → None

Bind the listener and event mask to the DomainParticipant.

property_subscription_reader

Get the subscription built-in topic reader.

unregister_contentfilter (*self*: rti.connexdds.DomainParticipant, *name*: str) → None

Unregister content filter previously registered to this DomainParticipant.

unregister_type (*self*: rti.connexdds.DomainParticipant, *name*: str) → None

Unregister a type previously registered to this DomainParticipant.

class rti.connexdds.DomainParticipantConfigParams

Bases: pybind11_object

DOMAIN_ID_USE_XML_CONFIG = -1

ENTITY_NAME_USE_XML_CONFIG =
'com.rti.dds.domain.entity_name_use_xml_config'

QOS_ELEMENT_NAME_USE_XML_CONFIG =
'com.rti.dds.domain.qos_element_name_use_xml_config'

__eq__ (*self*: rti.connexdds.DomainParticipantConfigParams, *arg0*: rti.connexdds.DomainParticipantConfigParams) → bool

Test for equality.

__hash__ = None

```

__init__ (self: rti.connexdds.DomainParticipantConfigParams, domain_id: int =
    DomainParticipantConfigParams.DOMAIN_ID_USE_XML_CONFIG, participant_name:
    str = DomainParticipantConfigParams.ENTITY_NAME_USE_XML_CONFIG,
    qos_library_name: str =
    DomainParticipantConfigParams.QOS_ELEMENT_NAME_USE_XML_CONFIG,
    qos_profile_name: str =
    DomainParticipantConfigParams.QOS_ELEMENT_NAME_USE_XML_CONFIG,
    domain_entity_qos_library_name: str =
    DomainParticipantConfigParams.QOS_ELEMENT_NAME_USE_XML_CONFIG,
    domain_entity_qos_profile_name: str =
    DomainParticipantConfigParams.QOS_ELEMENT_NAME_USE_XML_CONFIG) →
    None

```

Create a DomainParticipantConfigParams object with the specified values.

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.DomainParticipantConfigParams, arg0:
    rti.connexdds.DomainParticipantConfigParams) → bool

```

Test for inequality.

property domain_entity_qos_library_name

The QoS library name containing the QoS profile from which the all the entities defined under the participant configuraton are created.

property domain_entity_qos_profile_name

The QoS profile name from which the all the entities defined under the participant configuraton are created.

property domain_id

The domain id from which the DomainParticipant is created.

property participant_name

The name assigned to the DomainParticipant.

property participant_qos_library_name

The name of the library containing the DomainParticipant's QoS.

property participant_qos_profile_name

The name of the profile containing the DomainParticipant's QoS.

```
class rti.connexdds.DomainParticipantFactoryQos
```

```
Bases: pybind11_object
```

```
__eq__ (self: rti.connexdds.DomainParticipantFactoryQos, arg0:
    rti.connexdds.DomainParticipantFactoryQos) → bool

```

Test for equality

```
__hash__ = None
```

__init__ (*self*: rti.connextdds.DomainParticipantFactoryQos) → None

Create a DomainParticipantFactoryQos with the default value for each policy.

__lshift__ (**args*, ***kwargs*)

Overloaded function.

1. **__lshift__**(*self*: rti.connextdds.DomainParticipantFactoryQos, *arg0*: rti.connextdds.EntityFactory) -> rti.connextdds.DomainParticipantFactoryQos

Set the EntityFactoryQoS.

2. **__lshift__**(*self*: rti.connextdds.DomainParticipantFactoryQos, *arg0*: rti.connextdds.SystemResourceLimits) -> rti.connextdds.DomainParticipantFactoryQos

Set the SystemResourceLimitsQoS.

3. **__lshift__**(*self*: rti.connextdds.DomainParticipantFactoryQos, *arg0*: rti.connextdds.Monitoring) -> rti.connextdds.DomainParticipantFactoryQos

Set the MonitoringQoS.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.DomainParticipantFactoryQos, *arg0*: rti.connextdds.DomainParticipantFactoryQos) → bool

Test for inequality.

__rshift__ (**args*, ***kwargs*)

Overloaded function.

1. **__rshift__**(*self*: rti.connextdds.DomainParticipantFactoryQos, *arg0*: rti.connextdds.EntityFactory) -> rti.connextdds.DomainParticipantFactoryQos

Get the EntityFactoryQoS.

2. **__rshift__**(*self*: rti.connextdds.DomainParticipantFactoryQos, *arg0*: rti.connextdds.SystemResourceLimits) -> rti.connextdds.DomainParticipantFactoryQos

Get the SystemResourceLimitsQoS.

3. **__rshift__**(*self*: rti.connextdds.DomainParticipantFactoryQos, *arg0*: rti.connextdds.Monitoring) -> rti.connextdds.DomainParticipantFactoryQos

Get the MonitoringQoS.

__str__ (*self*: rti.connextdds.DomainParticipantFactoryQos) → str

property entity_factory

Get/set EntityFactory QoS.

property monitoring

Get/set Monitoring QoS.

property system_resource_limits

Get/set SystemResourceLimits QoS.


```
to_string (self: rti.connextdds.DomainParticipantFactoryQos, format:
           rti.connextdds.QosPrintFormat = QosPrintFormat(), base:
           Optional[rti.connextdds.DomainParticipantFactoryQos] = None, print_all: bool =
           False) → str
```

Convert QoS to string based on params.

```
class rti.connextdds.DomainParticipantListener
```

Bases: *PublisherListener*, *SubscriberListener*, *AnyTopicListener*

```
__init__ (self: rti.connextdds.DomainParticipantListener) → None
```

```
__module__ = 'rti.connextdds'
```

```
on_invalid_local_identity_status_advance_notice (self: rti.connextdds.DomainParticipantListener, arg0:
           rti.connextdds.DomainParticipant, arg1:
           rti.connextdds.InvalidLocalIdentityAdvanceNoticeStatus) →
           None
```

On invalid local identity status advance notice callback

```
class rti.connextdds.DomainParticipantProtocolStatus
```

Bases: *pybind11_object*

Protocol status of a DomainParticipant

```
__init__ (*args, **kwargs)
```

```
__module__ = 'rti.connextdds'
```

```
property corrupted_rtps_message_count
```

The number of corrupted RTPS messages detected by the domain participant.

```
property corrupted_rtps_message_count_change
```

The incremental change in the number of corrupted RTPS messages detected by the domain participant since the last time the status was read.

```
property last_corrupted_message_timestamp
```

The timestamp when the last corrupted RTPS message was detected by the domain participant.

```
class rti.connextdds.DomainParticipantQos
```

Bases: *pybind11_object*

```
__eq__ (self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.DomainParticipantQos)
        → bool
```

Test for equality

```
__hash__ = None
```

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.DomainParticipantQos) -> None`

Create a DomainParticipantQos with the default value for each policy.

2. `__init__(self: rti.connextdds.DomainParticipantQos, participant: rti.connextdds.DomainParticipant) -> None`

Create a DomainParticipantQos with settings equivalent to those of the provided DomainParticipant object.

3. `__init__(self: rti.connextdds.DomainParticipantQos, other: rti.connextdds.DomainParticipantQos) -> None`

Create a copy of a DomainParticipantQos object.

`__lshift__` (*args, **kwargs)

Overloaded function.

1. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.UserData) -> rti.connextdds.DomainParticipantQos`

Set the UserDataQoS.

2. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.EntityFactory) -> rti.connextdds.DomainParticipantQos`

Set the EntityFactoryQoS.

3. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.WireProtocol) -> rti.connextdds.DomainParticipantQos`

Set the WireProtocolQoS.

4. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.TransportBuiltin) -> rti.connextdds.DomainParticipantQos`

Set the TransportBuiltinQoS.

5. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Discovery) -> rti.connextdds.DomainParticipantQos`

Set the DiscoveryQoS.

6. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.DomainParticipantResourceLimits) -> rti.connextdds.DomainParticipantQos`

Set the DomainParticipantResourceLimitsQoS.

7. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Event) -> rti.connextdds.DomainParticipantQos`

Set the EventQoS.

8. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.ReceiverPool) -> rti.connextdds.DomainParticipantQos`

Set the ReceiverPoolQoS.

9. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Database) -> rti.connextdds.DomainParticipantQos`

Set the DatabaseQoS.

10. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.DiscoveryConfig) -> rti.connextdds.DomainParticipantQos`

Set the DiscoveryConfigQoS.

11. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Property) -> rti.connextdds.DomainParticipantQos`

Set the PropertyQoS.

12. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.EntityName) -> rti.connextdds.DomainParticipantQos`

Set the EntityNameQoS.

13. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.TransportMulticastMapping) -> rti.connextdds.DomainParticipantQos`

Set the TransportMulticastMappingQoS.

14. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Partition) -> rti.connextdds.DomainParticipantQos`

Set the PartitionQoS.

15. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.TransportUnicast) -> rti.connextdds.DomainParticipantQos`

Set the TransportUnicastQoS.

16. `__lshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Service) -> rti.connextdds.DomainParticipantQos`

Set the ServiceQoS.

```
__module__ = 'rti.connextdds'
```

```
__ne__ (self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.DomainParticipantQos)
    → bool
```

Test for inequality.

```
__rshift__ (*args, **kwargs)
```

Overloaded function.

1. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.UserData) -> rti.connextdds.DomainParticipantQos`

Get the UserDataQoS.

2. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.EntityFactory) -> rti.connextdds.DomainParticipantQos`

Get the EntityFactoryQoS.

3. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.WireProtocol) -> rti.connextdds.DomainParticipantQos`

Get the WireProtocolQoS.

4. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.TransportBuiltin) -> rti.connextdds.DomainParticipantQos`

Get the TransportBuiltinQoS.

5. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Discovery) -> rti.connextdds.DomainParticipantQos`

Get the DiscoveryQoS.

6. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.DomainParticipantResourceLimits) -> rti.connextdds.DomainParticipantQos`

Get the DomainParticipantResourceLimitsQoS.

7. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Event) -> rti.connextdds.DomainParticipantQos`

Get the EventQoS.

8. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.ReceiverPool) -> rti.connextdds.DomainParticipantQos`

Get the ReceiverPoolQoS.

9. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Database) -> rti.connextdds.DomainParticipantQos`

Get the DatabaseQoS.

10. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.DiscoveryConfig) -> rti.connextdds.DomainParticipantQos`

Get the DiscoveryConfigQoS.

11. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Property) -> rti.connextdds.DomainParticipantQos`

Get the PropertyQoS.

12. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.EntityName) -> rti.connextdds.DomainParticipantQos`

Get the EntityNameQoS.

13. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.TransportMulticastMapping) -> rti.connextdds.DomainParticipantQos`

Get the TransportMulticastMappingQoS.

14. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Partition) -> rti.connextdds.DomainParticipantQos`

Get the PartitionQoS.

15. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.TransportUnicast) -> rti.connextdds.DomainParticipantQos`

Get the TransportUnicastQoS.

16. `__rshift__(self: rti.connextdds.DomainParticipantQos, arg0: rti.connextdds.Service) -> rti.connextdds.DomainParticipantQos`

Get the ServiceQoS.

`__str__(self: rti.connextdds.DomainParticipantQos) → str`

property database

Get/set Database QoS.

property default_unicast

Get/set TransportUnicast QoS.

property discovery

Get/set Discovery QoS.

property discovery_config

Get/set DiscoveryConfig QoS.

property entity_factory

Get/set EntityFactory QoS.

property event

Get/set Event QoS.

property participant_name

Get/set EntityName QoS.

property partition

Get/set Partition QoS.

property property

Get/set Property QoS.

property receiver_pool

Get/set ReceiverPool QoS.

property resource_limits

Get/set DomainParticipantResourceLimits QoS.

property service

Get/set Service QoS.

`to_string(self: rti.connextdds.DomainParticipantQos, format: rti.connextdds.QosPrintFormat = QosPrintFormat(), base: Optional[rti.connextdds.DomainParticipantQos] = None, print_all: bool = False) → str`

Convert QoS to string based on params.

property transport_builtin

Get/set TransportBuiltin QoS.

property transport_multicast_mapping

Get/set TransportMulticastMapping QoS.

property user_data

Get/set UserData QoS.

property wire_protocol

Get/set WireProtocol QoS.

class rti.connexdds.DomainParticipantResourceLimits

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.DomainParticipantResourceLimits, *arg0*: rti.connexdds.DomainParticipantResourceLimits) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.DomainParticipantResourceLimits) → None

Create a default policy.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DomainParticipantResourceLimits, *arg0*: rti.connexdds.DomainParticipantResourceLimits) → bool

Test for inequality.

property channel_filter_expression_max_length

Maximum length of a channel ChannelSettings.filter_expression in a MultiChannel DataWriter.

property channel_seq_max_length

Maximum number of channels that can be specified in MultiChannel for MultiChannel DataWriters.

property content_filter_allocation

Allocation settings applied to content filter.

This property's getter returns a deep copy.

property content_filter_hash_buckets

Number of hash buckets for content filters.

property content_filtered_topic_allocation

Allocation settings applied to content filtered topic.

This property's getter returns a deep copy.

property content_filtered_topic_hash_buckets

Number of hash buckets for content filtered topics.

property contentfilter_property_max_length

This field is the maximum length of all data related to a Content-filtered topic.

property deserialized_type_object_dynamic_allocation_threshold

A threshold, in bytes, for dynamic memory allocation for the deserialized TypeObject.

property flow_controller_allocation

Allocation settings applied to flow controllers.

This property's getter returns a deep copy.

property flow_controller_hash_buckets

Number of hash buckets for flow controllers.

property ignored_entity_allocation

Allocation settings applied to ignored entities.

This property's getter returns a deep copy.

property ignored_entity_hash_buckets

Number of hash buckets for ignored entities.

property ignored_entity_replacement_kind

Replacement policy for the ignored entities. It sets what entity can be replaced when resource limits set in DomainParticipantResourceLimits.ignored_entity_allocation are reached.

property local_publisher_allocation

Allocation settings applied to local Publisher.

This property's getter returns a deep copy.

property local_publisher_hash_buckets

Number of hash buckets for local Publisher.

property local_reader_allocation

Allocation settings applied to local DataReaders.

This property's getter returns a deep copy.

property local_reader_hash_buckets

Hash buckets settings applied to local DataReaders.

property local_subscriber_allocation

Allocation settings applied to local Subscriber.

This property's getter returns a deep copy.

property local_subscriber_hash_buckets

Number of hash buckets for local Subscriber.

property local_topic_allocation

Allocation settings applied to local Topic.

This property's getter returns a deep copy.

property local_topic_hash_buckets

Number of hash buckets for local Topic.

property local_writer_allocation

Allocation settings applied to local DataWriters.

This property's getter returns a deep copy.

property local_writer_hash_buckets

Hash buckets settings applied to local DataWriters.

property matching_reader_writer_pair_allocation

Allocation settings applied to matching local reader and remote/local writer pairs.

This property's getter returns a deep copy.

property matching_reader_writer_pair_hash_buckets

Number of hash buckets for matching local reader and remote/local writer pairs.

property matching_writer_reader_pair_allocation

Allocation settings applied to matching local writer and remote/local reader pairs.

This property's getter returns a deep copy.

property matching_writer_reader_pair_hash_buckets

Number of hash buckets for matching local writer and remote/local reader pairs.

property max_endpoint_group_cumulative_characters

Maximum number of combined role_name characters allowed in all EndpointGroup in a Availability.

property max_endpoint_groups

Maximum number of EndpointGroup allowable in a Availability.

property max_gather_destinations

Maximum number of destinations per RTI Connext send.

property max_partition_cumulative_characters

Maximum number of combined characters allowable in all partition names in a Partition.

property max_partitions

Maximum number of partition name strings allowable in a Partition.

property outstanding_asynchronous_sample_allocation

Allocation settings applied to the maximum number of samples (from DataWriter) waiting to be asynchronously written.

This property's getter returns a deep copy.

property participant_property_list_max_length

Maximum number of properties associated with the DomainParticipant.

property participant_property_string_max_length

Maximum string length of the properties associated with the DomainParticipant.

property participant_user_data_max_length

Maximum length of user data in DomainParticipantQos and ParticipantBuiltinTopicData.

property publisher_group_data_max_length

Maximum length of group data in PublisherQos and PublicationBuiltinTopicData.

property query_condition_allocation

Allocation settings applied to query condition pool.

This property's getter returns a deep copy.

property read_condition_allocation

Allocation settings applied to read condition pool.

This property's getter returns a deep copy.

property reader_data_tag_list_max_length

Maximum number of data tags associated with a DataReader.

property reader_data_tag_string_max_length

Maximum string length of the data tags associated with a DataReader.

property reader_property_list_max_length

Maximum number of properties associated with a DataReader.

property reader_property_string_max_length

Maximum string length of the properties associated with a DataReader.

property reader_user_data_max_length

Maximum length of user data in DataReaderQos and SubscriptionBuiltinTopicData.

property remote_participant_allocation

Allocation settings applied to remote DomainParticipants.

This property's getter returns a deep copy.

property remote_participant_hash_buckets

Number of hash buckets for remote DomainParticipants.

property remote_reader_allocation

Allocation settings applied to remote DataReaders.

This property's getter returns a deep copy.

property remote_reader_hash_buckets

Number of hash buckets for remote DataReaders.

property remote_topic_query_allocation

Allocation settings applied to remote TopicQueries.

This property's getter returns a deep copy.

property remote_topic_query_hash_buckets

Number of hash buckets for remote TopicQueries.

property remote_writer_allocation

Allocation settings applied to remote DataWriters.

This property's getter returns a deep copy.

property remote_writer_hash_buckets

Number of hash buckets for remote DataWriters.

property serialized_type_object_dynamic_allocation_threshold

A threshold, in bytes, for dynamic memory allocation for the serialized TypeObject.

property shmem_ref_transfer_mode_max_segments

Maximum number of segments created by all DataWriters belonging to a DomainParticipant.

property subscriber_group_data_max_length

Maximum length of group data in SubscriberQos and SubscriptionBuiltinTopicData.

property topic_data_max_length

Maximum length of topic data in TopicQos, TopicBuiltinTopicData, PublicationBuiltinTopicData and SubscriptionBuiltinTopicData.

property transport_info_list_max_length

Maximum number of installed transports to send and receive information about in ParticipantBuiltinTopicData.transport_info.

property type_code_max_serialized_length

Maximum size of serialized string for type code.

property type_object_max_deserialized_length

The maximum number of bytes that a deserialized TypeObject can consume.

property type_object_max_serialized_length

The maximum length, in bytes, that the buffer to serialize a TypeObject can consume.

property writer_data_tag_list_max_length

Maximum number of data tags associated with a DataWriter.

property writer_data_tag_string_max_length

Maximum string length of the data tags associated with a DataWriter.

property writer_property_list_max_length

Maximum number of properties associated with a DataWriter.

property writer_property_string_max_length

Maximum string length of the properties associated with a DataWriter.

property writer_user_data_max_length

Maximum length of user data in DataWriterQos and PublicationBuiltinTopicData.

class `rti.connexdds.DomainParticipantSeq`Bases: `pybind11_object`**__add__** (*self*: `rti.connexdds.DomainParticipantSeq`, *arg0*: `rti.connexdds.DomainParticipantSeq`)
→ `rti.connexdds.DomainParticipantSeq`**__bool__** (*self*: `rti.connexdds.DomainParticipantSeq`) → `bool`

Check whether the list is nonempty

__contains__ (*self*: `rti.connexdds.DomainParticipantSeq`, *x*: `rti.connexdds.DomainParticipant`)
→ `bool`Return true the container contains *x***__delitem__** (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: `rti.connexdds.DomainParticipantSeq`, *arg0*: `int`) -> `None`Delete the list elements at index *i*2. **__delitem__**(*self*: `rti.connexdds.DomainParticipantSeq`, *arg0*: `slice`) -> `None`

Delete list elements using a slice object

__eq__ (*self*: `rti.connexdds.DomainParticipantSeq`, *arg0*: `rti.connexdds.DomainParticipantSeq`) →
`bool`**__getitem__** (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: `rti.connexdds.DomainParticipantSeq`, *s*: `slice`) -> `rti.connexdds.DomainParticipantSeq`

Retrieve list elements using a slice object

2. **__getitem__**(*self*: `rti.connexdds.DomainParticipantSeq`, *arg0*: `int`) -> `rti.connexdds.DomainParticipant`**__hash__** = `None`**__iadd__** (*self*: `rti.connexdds.DomainParticipantSeq`, *arg0*: `rti.connexdds.DomainParticipantSeq`)
→ `rti.connexdds.DomainParticipantSeq`**__imul__** (*self*: `rti.connexdds.DomainParticipantSeq`, *arg0*: `int`) →
`rti.connexdds.DomainParticipantSeq`**__init__** (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: `rti.connexdds.DomainParticipantSeq`) -> `None`2. **__init__**(*self*: `rti.connexdds.DomainParticipantSeq`, *arg0*: `rti.connexdds.DomainParticipantSeq`) -> `None`

Copy constructor

3. `__init__(self: rti.connextdds.DomainParticipantSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.DomainParticipantSeq) → Iterator`

`__len__(self: rti.connextdds.DomainParticipantSeq) → int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.DomainParticipantSeq, arg0: int) → rti.connextdds.DomainParticipantSeq`

`__ne__(self: rti.connextdds.DomainParticipantSeq, arg0: rti.connextdds.DomainParticipantSeq) → bool`

`__rmul__(self: rti.connextdds.DomainParticipantSeq, arg0: int) → rti.connextdds.DomainParticipantSeq`

`__setitem__(*args, **kwargs)`
Overloaded function.

- `__setitem__(self: rti.connextdds.DomainParticipantSeq, arg0: int, arg1: rti.connextdds.DomainParticipant) -> None`
- `__setitem__(self: rti.connextdds.DomainParticipantSeq, arg0: slice, arg1: rti.connextdds.DomainParticipantSeq) -> None`

Assign list elements using a slice object

`append(self: rti.connextdds.DomainParticipantSeq, x: rti.connextdds.DomainParticipant) → None`
Add an item to the end of the list

`clear(self: rti.connextdds.DomainParticipantSeq) → None`
Clear the contents

`count(self: rti.connextdds.DomainParticipantSeq, x: rti.connextdds.DomainParticipant) → int`
Return the number of times `x` appears in the list

`extend(*args, **kwargs)`
Overloaded function.

- `extend(self: rti.connextdds.DomainParticipantSeq, L: rti.connextdds.DomainParticipantSeq) -> None`
Extend the list by appending all the items in the given list
- `extend(self: rti.connextdds.DomainParticipantSeq, L: Iterable) -> None`
Extend the list by appending all the items in the given list

`insert(self: rti.connextdds.DomainParticipantSeq, i: int, x: rti.connextdds.DomainParticipant) → None`
Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connextdds.DomainParticipantSeq) -> rti.connextdds.DomainParticipant

Remove and return the last item

2. pop(self: rti.connextdds.DomainParticipantSeq, i: int) -> rti.connextdds.DomainParticipant

Remove and return the item at index i

remove (self: rti.connextdds.DomainParticipantSeq, x: rti.connextdds.DomainParticipant) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

rti.connextdds.DoubleSeq

alias of *Float64Seq*

rti.connextdds.DoubleType

alias of *Float64Type*

class rti.connextdds.Durability

Bases: pybind11_object

AUTO_WRITER_DEPTH = 0

__eq__ (self: rti.connextdds.Durability, arg0: rti.connextdds.Durability) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. __init__(self: rti.connextdds.Durability) -> None

Create an empty Durability QoS policy.

2. __init__(self: rti.connextdds.Durability, kind: rti.connextdds.DurabilityKind) -> None

Create a Durability QoS policy with the given kind.

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.Durability, arg0: rti.connextdds.Durability) → bool

Test for inequality.

property direct_communication

Get/set whether a DataReader should receive samples directly from a TRANSIENT or PERSISTENT DataWriter.

property kind

Get/set the Durability kind.

persistent = <rti.connextdds.Durability object>

property storage_settings

Configures durable writer history and durable reader state.

transient = <rti.connexdds.Durability object>

transient_local = <rti.connexdds.Durability object>

volatile = <rti.connexdds.Durability object>

property writer_depth

The number of samples a durable DataWriter will send to a late joining DataReader.

class rti.connexdds.DurabilityKind

Bases: pybind11_object

class DurabilityKind

Bases: pybind11_object

Members:

VOLATILE : [default] RTI Connex does not need to keep any samples of data instances on behalf of any DataReader that is unknown by the DataWriter at the time the instance is written.

In other words, RTI Connex will only attempt to provide the data to existing subscribers.

This option does not require RTI Persistence Service.

TRANSIENT_LOCAL : RTI Connex will attempt to keep some samples so that they can be delivered to any potential late-joining DataReader.

Which particular samples are kept depends on other QoS such as History and ResourceLimits. RTI Connex is only required to keep the data in memory of the DataWriter that wrote the data.

Data is not required to survive the DataWriter.

For this setting to be effective, you must also set the Reliability.kind to ReliabilityKind.RELIABLE.

This option does not require RTI Persistence Service.

TRANSIENT : RTI Connex will attempt to keep some samples so that they can be delivered to any potential late-joining DataReader.

Which particular samples are kept depends on other QoS such as History and ResourceLimits. RTI Connex is only required to keep the data in memory and not in permanent storage.

Data is not tied to the lifecycle of the DataWriter.

Data will survive the DataWriter.

This option requires RTI Persistence Service.

PERSISTENT : Data is kept on permanent storage, so that they can outlive a system session.

This option requires RTI Persistence Service.

```

PERSISTENT = <DurabilityKind.PERSISTENT: 3>
TRANSIENT = <DurabilityKind.TRANSIENT: 2>
TRANSIENT_LOCAL = <DurabilityKind.TRANSIENT_LOCAL: 1>
VOLATILE = <DurabilityKind.VOLATILE: 0>

__eq__ (self: object, other: object) → bool
__getstate__ (self: object) → int
__hash__ (self: object) → int
__index__ (self: rti.connextdds.DurabilityKind.DurabilityKind) → int
__init__ (self: rti.connextdds.DurabilityKind.DurabilityKind, value: int) → None
__int__ (self: rti.connextdds.DurabilityKind.DurabilityKind) → int
__members__ = {'PERSISTENT': <DurabilityKind.PERSISTENT: 3>,
                'TRANSIENT': <DurabilityKind.TRANSIENT: 2>, 'TRANSIENT_LOCAL':
                <DurabilityKind.TRANSIENT_LOCAL: 1>, 'VOLATILE':
                <DurabilityKind.VOLATILE: 0>}
__module__ = 'rti.connextdds'
__ne__ (self: object, other: object) → bool
__repr__ (self: object) → str
__setstate__ (self: rti.connextdds.DurabilityKind.DurabilityKind, state: int) → None
__str__ ()
    name(self: handle) -> str

property name
property value

```

```

PERSISTENT = <DurabilityKind.PERSISTENT: 3>
TRANSIENT = <DurabilityKind.TRANSIENT: 2>
TRANSIENT_LOCAL = <DurabilityKind.TRANSIENT_LOCAL: 1>
VOLATILE = <DurabilityKind.VOLATILE: 0>

__eq__ (self: rti.connextdds.DurabilityKind, arg0: rti.connextdds.DurabilityKind) → bool
    Apply operator to underlying enumerated values.
__ge__ (self: rti.connextdds.DurabilityKind, arg0: rti.connextdds.DurabilityKind) → bool
    Apply operator to underlying enumerated values.

```

__gt__ (*self*: rti.connexdds.DurabilityKind, *arg0*: rti.connexdds.DurabilityKind) → bool
Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DurabilityKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.DurabilityKind, *arg0*: rti.connexdds.DurabilityKind.DurabilityKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.DurabilityKind) → *rti.connexdds.DurabilityKind.DurabilityKind*

Int conversion.

__le__ (*self*: rti.connexdds.DurabilityKind, *arg0*: rti.connexdds.DurabilityKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.DurabilityKind, *arg0*: rti.connexdds.DurabilityKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DurabilityKind, *arg0*: rti.connexdds.DurabilityKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.DurabilityKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.DurabilityService

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.DurabilityService, *arg0*: rti.connexdds.DurabilityService) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DurabilityService) -> None

Creates the default policy.

2. **__init__**(*self*: rti.connexdds.DurabilityService, *service_cleanup_delay*: rti.connexdds.Duration, *history_kind*: rti.connexdds.HistoryKind, *history_depth*: int, *max_samples*: int, *max_instances*: int, *max_samples_per_instance*: int) -> None

Creates an instance with all the parameters set.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.DurabilityService, *arg0*: rti.connextdds.DurabilityService) → bool
Test for inequality.

property history_depth

Controls when the service is able to remove all information regarding a data instances.

property history_kind

Controls when the service is able to remove all information regarding a data instances.

property max_instances

Controls when the service is able to remove all information regarding a data instances.

property max_samples

Controls when the service is able to remove all information regarding a data instances.

property max_samples_per_instance

Controls when the service is able to remove all information regarding a data instances.

property service_cleanup_delay

Controls when the service is able to remove all information regarding a data instances.

class rti.connextdds.Duration

Bases: pybind11_object

__add__ (*self*: rti.connextdds.Duration, *arg0*: rti.connextdds.Duration) → *rti.connextdds.Duration*
Add two Duration objects.

__eq__ (*self*: rti.connextdds.Duration, *arg0*: rti.connextdds.Duration) → bool
Compare Duration objects for equality.

__float__ (*self*: rti.connextdds.Duration) → float
Floating point value of Duration in seconds.

__ge__ (*self*: rti.connextdds.Duration, *arg0*: rti.connextdds.Duration) → bool
Greater-than-or-equal comparison for Duration objects.

__gt__ (*self*: rti.connextdds.Duration, *arg0*: rti.connextdds.Duration) → bool
Greater than comparison for Duration objects.

__hash__ = None

__iadd__ (*self*: rti.connextdds.Duration, *arg0*: rti.connextdds.Duration) → *rti.connextdds.Duration*
Add a Duration to another.

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: rti.connextdds.Duration) -> None

Create a Duration of 0 seconds.

2. `__init__(self: rti.connextdds.Duration, seconds: int, nanoseconds: int) -> None`

Create a Duration of the specified seconds and nanoseconds.

3. `__init__(self: rti.connextdds.Duration, duration: datetime.timedelta) -> None`

Create a Duration from a `datetime.timedelta`.

4. `__init__(self: rti.connextdds.Duration, float_duration: float) -> None`

Create a Duration from a floating point duration in seconds.

5. `__init__(self: rti.connextdds.Duration, seconds: int) -> None`

Create a Duration from an integer number of seconds.

6. `__init__(self: rti.connextdds.Duration, time_tuple: Tuple[int, int]) -> None`

Create a Duration from a tuple of (sec, nanosec).

7. `__init__(self: rti.connextdds.Duration, other: rti.connextdds.Duration) -> None`

Create a copy of a Duration.

`__int__(self: rti.connextdds.Duration) -> int`

Integer value of Duration in microseconds.

`__isub__(self: rti.connextdds.Duration, arg0: rti.connextdds.Duration) -> rti.connextdds.Duration`

Subtract a Duration from another.

`__le__(self: rti.connextdds.Duration, arg0: rti.connextdds.Duration) -> bool`

Less-than-or-equal comparison for Duration object.

`__lt__(self: rti.connextdds.Duration, arg0: rti.connextdds.Duration) -> bool`

Less than comparison for Duration objects.

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.Duration, arg0: int) -> rti.connextdds.Duration`

Multiply a Duration by an unsigned int.

`__ne__(self: rti.connextdds.Duration, arg0: rti.connextdds.Duration) -> bool`

Determine if Duration objects are unequal.

`__rmul__(self: rti.connextdds.Duration, arg0: int) -> rti.connextdds.Duration`

Multiply a Duration by an unsigned int.

`__sub__(self: rti.connextdds.Duration, arg0: rti.connextdds.Duration) -> rti.connextdds.Duration`

Subtract two Duration objects.

`__truediv__(self: rti.connextdds.Duration, arg0: int) -> rti.connextdds.Duration`

Divide a Duration by an unsigned int.

`automatic = <rti.connextdds.Duration object>`

compare (*self*: rti.connextdds.Duration, *other*: rti.connextdds.Duration) → int
 Compare this Duration to another. Returns -1 if this Duration is less, 0 if they are equal, and 1 if this Duration is greater.

static from_microseconds (*microseconds*: int) → rti.connextdds.Duration
 Get a Duration from microseconds.

static from_milliseconds (*milliseconds*: int) → rti.connextdds.Duration
 Get a Duration from milliseconds.

static from_seconds (*seconds*: float) → rti.connextdds.Duration
 Get a Duration from seconds.

infinite = <rti.connextdds.Duration object>

property nanosec
 Get/set the number of nanoseconds in the Duration.

property sec
 Get/set the number of seconds in the Duration.

to_microseconds (*self*: rti.connextdds.Duration) → int
 Returns Duration in microseconds.

to_milliseconds (*self*: rti.connextdds.Duration) → int
 Returns Duration in milliseconds.

to_seconds (*self*: rti.connextdds.Duration) → float
 Return the Duration in seconds.

to_timedelta (*self*: rti.connextdds.Duration) → datetime.timedelta
 Return the datetime.timedelta conversion of this Duration.

zero = <rti.connextdds.Duration object>

class rti.connextdds.DynamicData

Bases: pybind11_object

class ContentFilter

Bases: ContentFilterBase

__init__ (*self*: rti.connextdds.DynamicData.ContentFilter) → None

__module__ = 'rti.connextdds'

compile (*self*: rti.connextdds.DynamicData.ContentFilter, *expression*: str, *parameters*: rti.connextdds.StringSeq, *type_code*: Optional[rti.connextdds.DynamicType], *type_class_name*: str, *old_compile_data*: Optional[object]) → Optional[object]

Compile an instance of the content filter according to the filter expression and parameters of the given data type.

evaluate (*self*: rti.connexdds.DynamicData.ContentFilter, *compile_data*: *Optional*[object], *sample*: rti.connexdds.DynamicData, *meta_data*: rti.connexdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

finalize (*self*: rti.connexdds.DynamicData.ContentFilter, *compile_data*: *Optional*[object]) → None

A previously compiled instance of the content filter is no longer in use and resources can now be cleaned up.

class ContentFilteredTopic

Bases: *ITopicDescription*, *IAnyTopic*

__eq__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopic, *arg0*: rti.connexdds.DynamicData.ContentFilteredTopic) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopic, *topic*: rti.connexdds.DynamicData.Topic, *name*: str, *contentfilter*: rti.connexdds.Filter) → None

Create a ContentFilteredTopic with a name and Filter.

2. **__init__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopic, *topic_description*: rti.connexdds.DynamicData.ITopicDescription) → None

Cast a TopicDescription to a ContentFilteredTopic.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopic, *arg0*: rti.connexdds.DynamicData.ContentFilteredTopic) → bool

Test for inequality.

append_to_expression_parameter (*self*: rti.connexdds.DynamicData.ContentFilteredTopic, *index*: int, *extension*: str) → None

Append the extension to the end of parameter at the provided index, separated by a comma.

property filter_expression

Get the filter expression

property filter_parameters

Get/set the filter parameters.

static find (*participant*: rti.connexdds.DomainParticipant, *name*: str) → *Optional*[rti.connexdds.DynamicData.ContentFilteredTopic]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

remove_from_expression_parameter (*self*: rti.connexdds.DynamicData.ContentFilteredTopic, *index*: int, *remove_term*: str) → None

Removes the specified term from the parameter at the provided index.

set_filter (*self*: rti.connexdds.DynamicData.ContentFilteredTopic, *arg0*: rti.connexdds.Filter) → None

Set the filter.

property topic

Get the underlying Topic.

class ContentFilteredTopicSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) → rti.connexdds.DynamicData.ContentFilteredTopicSeq

__bool__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *x*: rti.connexdds.DynamicData.ContentFilteredTopic) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *s*: slice) -> rti.connexdds.DynamicData.ContentFilteredTopicSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: int) -> rti.connexdds.DynamicData.ContentFilteredTopic

__hash__ = None

__iadd__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) → rti.connexdds.DynamicData.ContentFilteredTopicSeq

__imul__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: int) → *rti.connexdds.DynamicData.ContentFilteredTopicSeq*

__init__ (**args*, ***kwargs*)
 Overloaded function.
 1. **__init__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) -> None
 2. **__init__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) -> None
 Copy constructor
 3. **__init__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) → Iterator

__len__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: int) → *rti.connexdds.DynamicData.ContentFilteredTopicSeq*

__ne__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) → bool

__rmul__ (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: int) → *rti.connexdds.DynamicData.ContentFilteredTopicSeq*

__setitem__ (**args*, ***kwargs*)
 Overloaded function.
 1. **__setitem__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: int, *arg1*: rti.connexdds.DynamicData.ContentFilteredTopic) -> None
 2. **__setitem__**(*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *arg0*: slice, *arg1*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) -> None
 Assign list elements using a slice object

append (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *x*: rti.connexdds.DynamicData.ContentFilteredTopic) → None
 Add an item to the end of the list

clear (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq) → None
 Clear the contents

count (*self*: rti.connexdds.DynamicData.ContentFilteredTopicSeq, *x*: rti.connexdds.DynamicData.ContentFilteredTopic) → int
 Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)
 Overloaded function.

1. `extend(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, L: rti.connextdds.DynamicData.ContentFilteredTopicSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.DynamicData.ContentFilteredTopicSeq, *i*: int, *x*: rti.connextdds.DynamicData.ContentFilteredTopic) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq) -> rti.connextdds.DynamicData.ContentFilteredTopic`

Remove and return the last item

2. `pop(self: rti.connextdds.DynamicData.ContentFilteredTopicSeq, i: int) -> rti.connextdds.DynamicData.ContentFilteredTopic`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.DynamicData.ContentFilteredTopicSeq, *x*: rti.connextdds.DynamicData.ContentFilteredTopic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class DataReader

Bases: *IDataReader*

class Selector

Bases: `pybind11_object`

__init__ (*self*: rti.connextdds.DynamicData.DataReader.Selector, *datareader*: rti.connextdds.DynamicData.DataReader) → None

Create a Selector for a DataReader to read/take based on selected conditions

__module__ = 'rti.connextdds'

condition (*self*: rti.connextdds.DynamicData.DataReader.Selector, *condition*: rti.connextdds.IReadCondition) → *rti.connextdds.DynamicData.DataReader.Selector*

Select samples based on a ReadCondition.

content (*self*: rti.connextdds.DynamicData.DataReader.Selector, *query*: rti.connextdds.Query) → *rti.connextdds.DynamicData.DataReader.Selector*

Select samples based on a Query.

instance (*self*: rti.connextdds.DynamicData.DataReader.Selector, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.DynamicData.DataReader.Selector*

Select a specific instance to read/take.

max_samples (*self*: rti.connexdds.DynamicData.DataReader.Selector, *max*: int) →
rti.connexdds.DynamicData.DataReader.Selector

Limit the number of samples read/taken by the Select.

next_instance (*self*: rti.connexdds.DynamicData.DataReader.Selector, *previous*:
 rti.connexdds.InstanceHandle) →
rti.connexdds.DynamicData.DataReader.Selector

Select the instance after the specified instance to read/take.

read (*self*: rti.connexdds.DynamicData.DataReader.Selector) → list

Read copies of available samples (data and info) based on the Selector settings.

read_data (*self*: rti.connexdds.DynamicData.DataReader.Selector) → list

Read copies of available valid data based on the Selector settings.

read_loaned (*self*: rti.connexdds.DynamicData.DataReader.Selector) →
rti.connexdds.DynamicData.LoanedSamples

Take available samples (data and info) based on the Selector settings and return them in a loaned container.

state (*self*: rti.connexdds.DynamicData.DataReader.Selector, *state*:
 rti.connexdds.DataState) → *rti.connexdds.DynamicData.DataReader.Selector*

Select samples with a specified data state.

take (*self*: rti.connexdds.DynamicData.DataReader.Selector) → list

Take copies of available samples (data and info) based on the Selector settings.

take_data (*self*: rti.connexdds.DynamicData.DataReader.Selector) → list

Take copies of available valid data based on the Selector settings.

take_loaned (*self*: rti.connexdds.DynamicData.DataReader.Selector) →
rti.connexdds.DynamicData.LoanedSamples

Read available samples (data and info) based on the Selector settings and return them in a loaned container.

__enter__ (*self*: rti.connexdds.DynamicData.DataReader) →
rti.connexdds.DynamicData.DataReader

Enter a context for this DataReader, to be cleaned up on exiting context

__eq__ (*self*: rti.connexdds.DynamicData.DataReader, *arg0*:
 rti.connexdds.DynamicData.DataReader) → bool

Test for equality.

__exit__ (*self*: rti.connexdds.DynamicData.DataReader, *arg0*: object, *arg1*: object, *arg2*:
 object) → None

Exit the context for this DataReader, cleaning up resources.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connextdds.DynamicData.DataReader, topic: rti.connextdds.DynamicData.Topic) -> None

Create a DataReader in the implicit subscriber with default QoS.

2. **__init__**(self: rti.connextdds.DynamicData.DataReader, topic: rti.connextdds.DynamicData.Topic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.DynamicData.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader in the implicit subscriber with specific QoS and a listener.

3. **__init__**(self: rti.connextdds.DynamicData.DataReader, cft: rti.connextdds.DynamicData.ContentFilteredTopic) -> None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with default QoS.

4. **__init__**(self: rti.connextdds.DynamicData.DataReader, cft: rti.connextdds.DynamicData.ContentFilteredTopic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.DynamicData.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with specific QoS.

5. **__init__**(self: rti.connextdds.DynamicData.DataReader, subscriber: rti.connextdds.Subscriber, topic: rti.connextdds.DynamicData.Topic) -> None

Create a DataReader.

6. **__init__**(self: rti.connextdds.DynamicData.DataReader, subscriber: rti.connextdds.Subscriber, topic: rti.connextdds.DynamicData.Topic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.DynamicData.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader in a subscriber with specific QoS and a listener.

7. **__init__**(self: rti.connextdds.DynamicData.DataReader, subscriber: rti.connextdds.Subscriber, cft: rti.connextdds.DynamicData.ContentFilteredTopic) -> None

Create a DataReader with a ContentFilteredTopic in a subscriber with default QoS.

8. **__init__**(self: rti.connextdds.DynamicData.DataReader, subscriber: rti.connextdds.Subscriber, cft: rti.connextdds.DynamicData.ContentFilteredTopic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.DynamicData.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader with a ContentFilteredTopic in a subscriber with specific QoS.

9. **__init__**(self: rti.connextdds.DynamicData.DataReader, reader: rti.connextdds.IAnyDataReader) -> None

Get a typed DataReader from an AnyDataReader.

10. **__init__**(self: rti.connextdds.DynamicData.DataReader, entity: rti.connextdds.IEntity) -> None

Get a typed DataReader from an Entity.

__lshift__ (self: rti.connextdds.DynamicData.DataReader, arg0: rti.connextdds.DataReaderQos) → rti.connextdds.DynamicData.DataReader

Set the DataReaderQos for this DataReader.

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.DynamicData.DataReader, arg0: rti.connextdds.DynamicData.DataReader) → bool

Test for inequality.

__rshift__ (*self*: rti.connextdds.DynamicData.DataReader, *arg0*: rti.connextdds.DataReaderQos) → *rti.connextdds.DynamicData.DataReader*

Get the DataReaderQos from this DataReader

acknowledge_all (**args*, ***kwargs*)

Overloaded function.

1. `acknowledge_all(self: rti.connextdds.DynamicData.DataReader) -> None`

Acknowledge all previously accessed samples.

2. `acknowledge_all(self: rti.connextdds.DynamicData.DataReader, arg0: rti.connextdds.AckResponseData) -> None`

Acknowledge all previously accessed samples.

acknowledge_sample (**args*, ***kwargs*)

Overloaded function.

1. `acknowledge_sample(self: rti.connextdds.DynamicData.DataReader, sample_info: rti.connextdds.SampleInfo) -> None`

Acknowledge a single sample.

2. `acknowledge_sample(self: rti.connextdds.DynamicData.DataReader, sample_info: rti.connextdds.SampleInfo, ack_response_data: rti.connextdds.AckResponseData) -> None`

Acknowledge a single sample with ack response data.

close (*self*: rti.connextdds.DynamicData.DataReader) → None

Close this DataReader.

property datareader_cache_status

Get the DataReaderCacheStatus for the DataReader.

property datareader_protocol_status

Get the DataReaderProtocolStatus for the DataReader.

property default_filter_state

Returns the filter state for the read/take operations.

static find_all_by_topic (*subscriber*: rti.connextdds.Subscriber, *topic_name*: str) → *rti.connextdds.DynamicData.DataReaderSeq*

Retrieve all DataReaders for the given topic name in the subscriber.

static find_by_name (**args*, ***kwargs*)

Overloaded function.

1. `find_by_name(participant: rti.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.DynamicData.DataReader]`

Find DataReader in DomainParticipant with the DataReader's name, returning the first found.

2. `find_by_name(subscriber: rti.connextdds.Subscriber, name: str) -> Optional[rti.connextdds.DynamicData.DataReader]`

Find DataReader in Subscriber with the DataReader's name, returning the first found.

static find_by_topic (*subscriber*: rti.connextdds.Subscriber, *name*: str) → Optional[*rti.connextdds.DynamicData.DataReader*]
Find DataReader in Subscriber with a topic name, returning the first found.

is_matched_publication_alive (*self*: rti.connextdds.DynamicData.DataReader, *arg0*: rti.connextdds.InstanceHandle) → bool
Check if a matched publication is alive.

key_value (*self*: rti.connextdds.DynamicData.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.DynamicData*
Retrieve the instance key that corresponds to an instance handle.

property listener
Gets or sets the listener with StatusMask.ALL

property liveliness_changed_status
Get the LivelinessChangedStatus for this DataReader.

lookup_instance (*self*: rti.connextdds.DynamicData.DataReader, *key_holder*: rti.connextdds.DynamicData) → *rti.connextdds.InstanceHandle*
Retrieve the instance handle that corresponds to an instance key_holder

matched_publication_data (*self*: rti.connextdds.DynamicData.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.PublicationBuiltinTopicData*
Get the PublicationBuiltinTopicData for a publication matched to this DataReader.

matched_publication_datareader_protocol_status (*self*: rti.connextdds.DynamicData.DataReader, *publication_handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.DataReaderProtocolStatus*
Get the DataReaderProtocolStatus for the DataReader based on the matched publication handle.

matched_publication_participant_data (*self*: rti.connextdds.DynamicData.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.ParticipantBuiltinTopicData*
Get the ParticipantBuiltinTopicData for a publication matched to this DataReader.

property matched_publications
Get a copy of the list of the currently matched publication handles.

property qos
The DataReaderQos for this DataReader.
This property's getter returns a deep copy.

read (*self*: rti.connexdds.DynamicData.DataReader) → list
 Read copies of all available samples (data and info).

read_data (*self*: rti.connexdds.DynamicData.DataReader) → list
 Read copies of all available valid data.

read_loaned (*self*: rti.connexdds.DynamicData.DataReader) →
rti.connexdds.DynamicData.LoanedSamples
 Read all available samples (data and info) and return them in a loaned container.

property requested_deadline_missed_status
 Get the RequestedDeadlineMissed status for the DataReader

property requested_incompatible_qos_status
 Get the RequestedIncompatibleQosStatus for the DataReader.

property sample_lost_status
 Get the SampleLostStatus for the DataReader.

property sample_rejected_status
 Get the SampleRejectedStatus for the DataReader.

select (*self*: rti.connexdds.DynamicData.DataReader) →
 dds::sub::DataReader<rti::core::xtypes::DynamicDataImpl,
 rti::sub::DataReaderImpl>::Selector
 Get a Selector to perform complex data selections, such as per-instance selection, content, and status filtering.

set_listener (*self*: rti.connexdds.DynamicData.DataReader, *listener*:
 rti.connexdds.DynamicData.DataReaderListener, *event_mask*:
 rti.connexdds.StatusMask) → None
 Set the listener and associated event mask.

property subscriber
 Returns the parent Subscriber of the DataReader.

property subscription_matched_status
 Get the SubscriptionMatchedStatus for the DataReader.

take (*self*: rti.connexdds.DynamicData.DataReader) → list
 Take copies of all available samples (data and info).

take_async (*condition*: *Optional*[ReadCondition] = None)

take_data (*self*: rti.connexdds.DynamicData.DataReader) → list
 Take copies of all available valid data.

take_data_async (*condition*: *Optional*[ReadCondition] = None)

take_loaned (*self*: rti.connexdds.DynamicData.DataReader) →
rti.connexdds.DynamicData.LoanedSamples
 Take all available samples (data and info) and return them in a loaned container.

property topic_description

Returns the TopicDescription associated with the DataReader.

property topic_name

Get the topic name associated with this DataReader.

property type_name

Get the type name associated with this DataReader.

wait_for_historical_data (*self*: rti.connextdds.DynamicData.DataReader, *max_wait*: rti.connextdds.Duration) → None

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind.

wait_for_historical_data_async (*self*: rti.connextdds.DynamicData.DataReader, *max_wait*: rti.connextdds.Duration) → object

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind. This call is awaitable and only for use with asyncio.

class DataReaderListener

Bases: pybind11_object

__init__ (*self*: rti.connextdds.DynamicData.DataReaderListener) → None

__module__ = 'rti.connextdds'

on_data_available (*self*: rti.connextdds.DynamicData.DataReaderListener, *arg0*: rti.connextdds.DynamicData.DataReader) → None

Data available callback.

on_liveliness_changed (*self*: rti.connextdds.DynamicData.DataReaderListener, *arg0*: rti.connextdds.DynamicData.DataReader, *arg1*: rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connextdds.DynamicData.DataReaderListener, *arg0*: rti.connextdds.DynamicData.DataReader, *arg1*: rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connextdds.DynamicData.DataReaderListener, *arg0*: rti.connextdds.DynamicData.DataReader, *arg1*: rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connextdds.DynamicData.DataReaderListener, *arg0*: rti.connextdds.DynamicData.DataReader, *arg1*: rti.connextdds.SampleLostStatus) → None

Sample lost callback.

on_sample_rejected (*self*: rti.connextdds.DynamicData.DataReaderListener, *arg0*: rti.connextdds.DynamicData.DataReader, *arg1*: rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connextdds.DynamicData.DataReaderListener, *arg0*: rti.connextdds.DynamicData.DataReader, *arg1*: rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class DataReaderSeq

Bases: pybind11_object

__add__ (*self*: rti.connextdds.DynamicData.DataReaderSeq, *arg0*: rti.connextdds.DynamicData.DataReaderSeq) → rti.connextdds.DynamicData.DataReaderSeq

__bool__ (*self*: rti.connextdds.DynamicData.DataReaderSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connextdds.DynamicData.DataReaderSeq, *x*: rti.connextdds.DynamicData.DataReader) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connextdds.DynamicData.DataReaderSeq, *arg0*: int) -> None

Delete the list elements at index i

2. **__delitem__**(*self*: rti.connextdds.DynamicData.DataReaderSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connextdds.DynamicData.DataReaderSeq, *arg0*: rti.connextdds.DynamicData.DataReaderSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connextdds.DynamicData.DataReaderSeq, *s*: slice) -> rti.connextdds.DynamicData.DataReaderSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connextdds.DynamicData.DataReaderSeq, *arg0*: int) -> rti.connextdds.DynamicData.DataReader

__hash__ = None

__iadd__ (*self*: rti.connexdds.DynamicData.DataReaderSeq, *arg0*: rti.connexdds.DynamicData.DataReaderSeq) → *rti.connexdds.DynamicData.DataReaderSeq*

__imul__ (*self*: rti.connexdds.DynamicData.DataReaderSeq, *arg0*: int) → *rti.connexdds.DynamicData.DataReaderSeq*

__init__ (**args*, ***kwargs*)
 Overloaded function.
 1. **__init__**(*self*: rti.connexdds.DynamicData.DataReaderSeq) -> None
 2. **__init__**(*self*: rti.connexdds.DynamicData.DataReaderSeq, *arg0*: rti.connexdds.DynamicData.DataReaderSeq) -> None
 Copy constructor
 3. **__init__**(*self*: rti.connexdds.DynamicData.DataReaderSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.DynamicData.DataReaderSeq) → Iterator

__len__ (*self*: rti.connexdds.DynamicData.DataReaderSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.DynamicData.DataReaderSeq, *arg0*: int) → *rti.connexdds.DynamicData.DataReaderSeq*

__ne__ (*self*: rti.connexdds.DynamicData.DataReaderSeq, *arg0*: rti.connexdds.DynamicData.DataReaderSeq) → bool

__rmul__ (*self*: rti.connexdds.DynamicData.DataReaderSeq, *arg0*: int) → *rti.connexdds.DynamicData.DataReaderSeq*

__setitem__ (**args*, ***kwargs*)
 Overloaded function.
 1. **__setitem__**(*self*: rti.connexdds.DynamicData.DataReaderSeq, *arg0*: int, *arg1*: rti.connexdds.DynamicData.DataReader) -> None
 2. **__setitem__**(*self*: rti.connexdds.DynamicData.DataReaderSeq, *arg0*: slice, *arg1*: rti.connexdds.DynamicData.DataReaderSeq) -> None
 Assign list elements using a slice object

append (*self*: rti.connexdds.DynamicData.DataReaderSeq, *x*: rti.connexdds.DynamicData.DataReader) → None
 Add an item to the end of the list

clear (*self*: rti.connexdds.DynamicData.DataReaderSeq) → None
 Clear the contents

count (*self*: rti.connexdds.DynamicData.DataReaderSeq, *x*: rti.connexdds.DynamicData.DataReader) → int
 Return the number of times *x* appears in the list

extend (*args, **kwargs)

Overloaded function.

1. extend(self: rti.connextdds.DynamicData.DataReaderSeq, L: rti.connextdds.DynamicData.DataReaderSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.DynamicData.DataReaderSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.DynamicData.DataReaderSeq, i: int, x: rti.connextdds.DynamicData.DataReader) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connextdds.DynamicData.DataReaderSeq) -> rti.connextdds.DynamicData.DataReader

Remove and return the last item

2. pop(self: rti.connextdds.DynamicData.DataReaderSeq, i: int) -> rti.connextdds.DynamicData.DataReader

Remove and return the item at index i

remove (self: rti.connextdds.DynamicData.DataReaderSeq, x: rti.connextdds.DynamicData.DataReader) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class DataWriter

Bases: *IEntity, IAnyDataWriter*

__enter__ (self: rti.connextdds.DynamicData.DataWriter) → *rti.connextdds.DynamicData.DataWriter*

Enter a context for this DataWriter, to be cleaned up on exiting context

__eq__ (self: rti.connextdds.DynamicData.DataWriter, arg0: rti.connextdds.DynamicData.DataWriter) → bool

Test for equality.

__exit__ (self: rti.connextdds.DynamicData.DataWriter, arg0: object, arg1: object, arg2: object) → None

Exit the context for this DataWriter, cleaning up resources.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. __init__(self: rti.connextdds.DynamicData.DataWriter, topic: rti.connextdds.DynamicData.Topic) -> None

Creates a DataWriter in the implicit publisher with default QoS.

2. __init__(self: rti.connextdds.DynamicData.DataWriter, topic: rti.connextdds.DynamicData.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.DynamicData.DataReaderSeq) -> None

Data.DataWriterListener = None, mask: rti.connexdds.StatusMask = StatusMask.ALL)
-> None

Creates a DataWriter in the implicit publisher with specific QoS and optionally a listener.

3. `__init__(self: rti.connexdds.DynamicData.DataWriter, pub: rti.connexdds.Publisher, topic: rti.connexdds.DynamicData.Topic)` -> None

Creates a DataWriter in a publisher with default QoS.

4. `__init__(self: rti.connexdds.DynamicData.DataWriter, pub: rti.connexdds.Publisher, topic: rti.connexdds.DynamicData.Topic, qos: rti.connexdds.DataWriterQos, listener: rti.connexdds.DynamicData.DataWriterListener = None, mask: rti.connexdds.StatusMask = StatusMask.ALL)` -> None

Creates a DataWriter in a publisher with specific QoS and optionally a listener.

5. `__init__(self: rti.connexdds.DynamicData.DataWriter, writer: rti.connexdds.IAnyDataWriter)` -> None

Create a typed DataWriter from an AnyDataWriter.

6. `__init__(self: rti.connexdds.DynamicData.DataWriter, entity: rti.connexdds.IEntity)` -> None

Create a typed DataWriter from an Entity.

`__lshift__ (*args, **kwargs)`

Overloaded function.

1. `__lshift__(self: rti.connexdds.DynamicData.DataWriter, arg0: rti.connexdds.DataWriterQos)` -> `rti.connexdds.DynamicData.DataWriter`

Sets the DataWriterQos.

2. `__lshift__(self: rti.connexdds.DynamicData.DataWriter, arg0: Tuple[rti.connexdds.DynamicData, rti.connexdds.Time])` -> `rti.connexdds.DynamicData.DataWriter`

Writes a paired sample with a timestamp.

3. `__lshift__(self: rti.connexdds.DynamicData.DataWriter, arg0: Tuple[rti.connexdds.DynamicData, rti.connexdds.InstanceHandle])` -> `rti.connexdds.DynamicData.DataWriter`

Writes a paired sample with an instance handle.

4. `__lshift__(self: rti.connexdds.DynamicData.DataWriter, arg0: rti.connexdds.DynamicData.TimestampedSeq)` -> `rti.connexdds.DynamicData.DataWriter`

Writes a sequence of pairs of samples with timestamps.

5. `__lshift__(self: rti.connexdds.DynamicData.DataWriter, arg0: rti.connexdds.DynamicDataSeq)` -> `rti.connexdds.DynamicData.DataWriter`

Writes a sequence of samples.

6. `__lshift__(self: rti.connexdds.DynamicData.DataWriter, arg0: rti.connexdds.DynamicData)` -> `rti.connexdds.DynamicData.DataWriter`

Writes a sample.

`__module__ = 'rti.connexdds'`

`__ne__(self: rti.connexdds.DynamicData.DataWriter, arg0: rti.connexdds.DynamicData.DataWriter)` → bool

Test for inequality.

`__rshift__(self: rti.connexdds.DynamicData.DataWriter, arg0: rti.connexdds.DataWriterQos)` → `rti.connexdds.DynamicData.DataWriter`

Get the DataWriterQos.

assert_liveliness (*self*: rti.connexdds.DynamicData.DataWriter) → None

Manually asserts the liveliness of the DataWriter.

close (*self*: rti.connexdds.DynamicData.DataWriter) → None

Close this DataWriter.

create_data (*self*: rti.connexdds.DynamicData.DataWriter) → *rti.connexdds.DynamicData*

Create data of the writer's associated type and initialize it.

property datawriter_cache_status

Get a copy of the cache status for this writer.

property datawriter_protocol_status

Get a copy of the protocol status for this writer.

dispose_instance (**args*, ***kwargs*)

Overloaded function.

1. `dispose_instance(self: rti.connexdds.DynamicData.DataWriter, handle: rti.connexdds.InstanceHandle) -> rti.connexdds.DynamicData.DataWriter`

Dispose an instance.

2. `dispose_instance(self: rti.connexdds.DynamicData.DataWriter, handle: rti.connexdds.InstanceHandle, timestamp: rti.connexdds.Time) -> rti.connexdds.DynamicData.DataWriter`

Dispose an instance with a timestamp.

3. `dispose_instance(self: rti.connexdds.DynamicData.DataWriter, params: rti.connexdds.WriteParams) -> None`

Dispose an instance with params.

dispose_instance_async (**args*, ***kwargs*)

Overloaded function.

1. `dispose_instance_async(self: rti.connexdds.DynamicData.DataWriter, handle: rti.connexdds.InstanceHandle) -> object`

Dispose an instance.

2. `dispose_instance_async(self: rti.connexdds.DynamicData.DataWriter, handle: rti.connexdds.InstanceHandle, timestamp: rti.connexdds.Time) -> object`

Dispose an instance with a timestamp.

3. `dispose_instance_async(self: rti.connexdds.DynamicData.DataWriter, key_holder: rti.connexdds.DynamicData) -> object`

Dispose the instance associated with key_holder.

4. `dispose_instance_async(self: rti.connexdds.DynamicData.DataWriter, key_holder: rti.connexdds.DynamicData, timestamp: rti.connexdds.Time) -> object`

Dispose the instance associated with key_holder using a timestamp

5. `dispose_instance_async(self: rti.connexdds.DynamicData.DataWriter, params: rti.connexdds.WriteParams) -> object`

Dispose an instance with params.

static find_all_by_topic (*publisher*: rti.connextdds.Publisher, *topic_name*: str) → *rti.connextdds.DynamicData.DataWriterSeq*

Retrieve all DataWriters for the given topic name in the publisher.

static find_by_name (**args*, ***kwargs*)

Overloaded function.

1. **find_by_name**(*participant*: rti.connextdds.DomainParticipant, *name*: str) -> Optional[*rti.connextdds.DynamicData.DataWriter*]

Find DataWriter in DomainParticipant with the provided name, returning the first found.

2. **find_by_name**(*publisher*: rti.connextdds.Publisher, *name*: str) -> Optional[*rti.connextdds.DynamicData.DataWriter*]

Find DataWriter in Publisher with the DataReader's name, returning the first found.

static find_by_topic (*publisher*: rti.connextdds.Publisher, *name*: str) → Optional[*rti.connextdds.DynamicData.DataWriter*]

Find DataWriter in publisher with a topic name, returning the first found.

flush (*self*: rti.connextdds.DynamicData.DataWriter) → None

Flushes the batch in progress in the context of thecalling thread.

is_matched_subscription_active (*self*: rti.connextdds.DynamicData.DataWriter, *arg0*: rti.connextdds.InstanceHandle) → bool

A boolean indicating whether or not the matched subscription is active.

is_sample_app_acknowledged (*self*: rti.connextdds.DynamicData.DataWriter, *sample_id*: rti.connextdds.SampleIdentity) → bool

Indicates if a sample is considered application-acknowledged.

key_value (*self*: rti.connextdds.DynamicData.DataWriter, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.DynamicData*

Retrieve the instance key that corresponds to an instance handle.

property listener

Get the listener associated with the DataWriter or set the listener.

property liveliness_lost_status

Get a copy of the LivelinessLostStatus.

lookup_instance (*self*: rti.connextdds.DynamicData.DataWriter, *key_holder*: rti.connextdds.DynamicData) → *rti.connextdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

matched_subscription_data (*self*: rti.connextdds.DynamicData.DataWriter, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.SubscriptionBuiltinTopicData*

Get the SubscriptionBuiltinTopicData for a subscription matched to this DataWriter.

matched_subscription_datawriter_protocol_status (**args*, ***kwargs*)

Overloaded function.

1. `matched_subscription_datawriter_protocol_status(self: rti.connextdds.DynamicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.DataWriterProtocolStatus`

Get a copy of the protocol status for this writer per a matched subscription handle.

2. `matched_subscription_datawriter_protocol_status(self: rti.connextdds.DynamicData.DataWriter, locator: rti.connextdds.Locator) -> rti.connextdds.DataWriterProtocolStatus`

Get a copy of the protocol status for this writer per a matched subscription locator.

matched_subscription_participant_data (*self: rti.connextdds.DynamicData.DataWriter, handle: rti.connextdds.InstanceHandle*) → *rti.connextdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a subscription matched to this DataWriter.

property matched_subscriptions

Get a copy of the list of the currently matched subscription handles.

property matched_subscriptions_locators

The locators used to communicate with matched DataReaders.

property offered_deadline_missed_status

Get a copy of the OfferedDeadlineMissedStatus.

property offered_incompatible_qos_status

Get a copy of the OfferedIncompatibleQosStatus

property publication_matched_status

Get a copy of the PublicationMatchedStatus

property publisher

Get the Publisher that owns this DataWriter.

property qos

The DataWriterQos for this DataWriter. This property's getter returns a deep copy.

register_instance (**args, **kwargs*)

Overloaded function.

1. `register_instance(self: rti.connextdds.DynamicData.DataWriter, key_holder: rti.connextdds.DynamicData) -> rti.connextdds.InstanceHandle`

Informs RTI Connex that the application will be modifying a particular instance.

2. `register_instance(self: rti.connextdds.DynamicData.DataWriter, key_holder: rti.connextdds.DynamicData, timestamp: rti.connextdds.Time) -> rti.connextdds.InstanceHandle`

Informs RTI Connex that the application will be modifying a particular instance and specified the timestamp.

3. `register_instance(self: rti.connextdds.DynamicData.DataWriter, key_holder: rti.connextdds.DynamicData, params: rti.connextdds.WriteParams) -> rti.connextdds.InstanceHandle`

Registers instance with parameters.

property reliable_reader_activity_changed_status

Get a copy of the reliable reader activity changed status for this writer.

property reliable_writer_cache_changed_status

Get a copy of the reliable cache status for this writer.

property service_request_accepted_status

Get a copy of the service request accepted status for this writer.

set_listener (*self*: rti.connextdds.DynamicData.DataWriter, *listener*: rti.connextdds.DynamicData.DataWriterListener, *event_mask*: rti.connextdds.StatusMask) → None

Set the listener and event mask for the DataWriter.

property topic

Get the Topic object associated with this DataWriter.

property topic_name

Get the topic name associated with this DataWriter.

property type_name

Get the type name for the topic object associated with this DataWriter.

unregister_instance (**args*, ***kwargs*)

Overloaded function.

1. `unregister_instance(self: rti.connextdds.DynamicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.DynamicData.DataWriter`

Unregister an instance.

2. `unregister_instance(self: rti.connextdds.DynamicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.DynamicData.DataWriter`

Unregister an instance with timestamp.

3. `unregister_instance(self: rti.connextdds.DynamicData.DataWriter, params: rti.connextdds.WriteParams) -> None`

Unregister an instance with parameters.

unregister_instance_async (**args*, ***kwargs*)

Overloaded function.

1. `unregister_instance_async(self: rti.connextdds.DynamicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> object`

Unregister an instance.

2. `unregister_instance_async(self: rti.connextdds.DynamicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Unregister an instance with timestamp.

3. `unregister_instance_async(self: rti.connextdds.DynamicData.DataWriter, key_holder: rti.connextdds.DynamicData) -> object`

Unregister the instance associated with `key_holder`.

4. `unregister_instance_async(self: rti.connextdds.DynamicData.DataWriter, key_holder: rti.connextdds.DynamicData, timestamp: rti.connextdds.Time) -> object`

Unregister the instance associate with `key_holder` using a timestamp.

5. `unregister_instance_async(self: rti.connextdds.DynamicData.DataWriter, params: rti.connextdds.WriteParams) -> object`

Unregister an instance with parameters.

wait_for_acknowledgments (*self*: rti.connextdds.DynamicData.DataWriter, *max_wait*: rti.connextdds.Duration) → None

Blocks the calling thread until all data written by a reliable DataWriter is acknowledged or until the timeout expires.

wait_for_asynchronous_publishing (*self*: rti.connextdds.DynamicData.DataWriter, *max_wait*: rti.connextdds.Duration) → None

This operation blocks the calling thread (up to `max_wait`) until all data written by the asynchronous DataWriter is sent and acknowledged (if reliable) by all matched DataReader entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that `max_wait` elapsed before all the data was sent and/or acknowledged.

In other words, this guarantees that sending to best effort DataReader is complete in addition to what `DataWriter.wait_for_acknowledgments()` provides.

If the DataWriter does not have `PublishMode` kind set to `PublishModeKind.ASYNCHRONOUS` the operation will complete immediately

wait_for_asynchronous_publishing_async (*self*: rti.connextdds.DynamicData.DataWriter, *max_wait*: rti.connextdds.Duration) → object

This function is awaitable until either a timeout of `max_wait` or all data written by the asynchronous DataWriter is sent and acknowledged (if reliable) by all matched DataReader entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that `max_wait` elapsed before all the data was sent and/or acknowledged. This function works with `asyncio`.

In other words, this guarantees that sending to best effort DataReader is complete in addition to what `DataWriter.wait_for_acknowledgments()` provides.

If the DataWriter does not have `PublishMode` kind set to `PublishModeKind.ASYNCHRONOUS` the operation will complete immediately

write (**args*, ***kwargs*)

Overloaded function.

1. `write(self: rti.connextdds.DynamicData.DataWriter, samples: rti.connextdds.DynamicDataSeq) -> None`

Write a sequence of samples.

2. `write(self: rti.connextdds.DynamicData.DataWriter, samples: rti.connextdds.DynamicDataSeq, timestamp: rti.connextdds.Time) -> None`

Write a sequence of samples with a timestamp.

3. `write(self: rti.connextdds.DynamicData.DataWriter, sample: rti.connextdds.DynamicData) -> None`

Write a sample.

4. `write(self: rti.connextdds.DynamicData.DataWriter, sample: rti.connextdds.DynamicData, timestamp: rti.connextdds.Time) -> None`

Write a sample with a specified timestamp.

5. `write(self: rti.connextdds.DynamicData.DataWriter, sample: rti.connextdds.DynamicData, handle: rti.connextdds.InstanceHandle) -> None`

Write a sample with an instance handle.

6. `write(self: rti.connextdds.DynamicData.DataWriter, sample: rti.connextdds.DynamicData, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> None`

Write a sample with an instance handle and specified timestamp.

7. `write(self: rti.connextdds.DynamicData.DataWriter, sample: rti.connextdds.DynamicData, params: rti.connextdds.WriteParams) -> None`

Write with advanced parameters.

8. `write(self: rti.connextdds.DynamicData.DataWriter, sample_data: dict) -> None`

Create a DynamicData object and write it with the given dictionary containing field names as keys.

write_async (*args, **kwargs)

Overloaded function.

1. `write_async(self: rti.connextdds.DynamicData.DataWriter, sample: rti.connextdds.DynamicData) -> object`

Write a sample. This method is awaitable and is only for use with asyncio.

2. `write_async(self: rti.connextdds.DynamicData.DataWriter, sample: rti.connextdds.DynamicData, timestamp: rti.connextdds.Time) -> object`

Write a sample with a specified timestamp. This methods is awaitable and only for use with asyncio.

3. `write_async(self: rti.connextdds.DynamicData.DataWriter, sample: rti.connextdds.DynamicData, handle: rti.connextdds.InstanceHandle) -> object`

Write a sample with an instance handle. This method is awaitable and only for use with asyncio.

4. `write_async(self: rti.connextdds.DynamicData.DataWriter, sample: rti.connextdds.DynamicData, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Write a sample with an instance handle and specified timestamp. This method is awaitable and only for use with asyncio.

5. `write_async(self: rti.connextdds.DynamicData.DataWriter, samples: rti.connextdds.DynamicDataSeq) -> object`

Write a sequence of samples. This method is awaitable and only for use with asyncio.

6. `write_async(self: rti.connextdds.DynamicData.DataWriter, samples: rti.connextdds.DynamicDataSeq, timestamp: rti.connextdds.Time) -> object`

Write a sequence of samples with a timestamp. This method is awaitable and only for use with asyncio.

7. `write_async(self: rti.connextdds.DynamicData.DataWriter, samples: rti.connextdds.DynamicDataSeq, handles: rti.connextdds.InstanceHandleSeq) -> object`

Write a sequence of samples with their instance handles. This method is awaitable and only for use with asyncio.

8. `write_async(self: rti.connextdds.DynamicData.DataWriter, samples: rti.connextdds.DynamicDataSeq, handles: rti.connextdds.InstanceHandleSeq, timestamp: rti.connextdds.Time) -> object`

Write a sequence of samples with their instance handles and a timestamp. This method is awaitable and only for use with asyncio.

9. `write_async(self: rti.connextdds.DynamicData.DataWriter, sample: rti.connextdds.DynamicData, params: rti.connextdds.WriteParams) -> object`

Write with advanced parameters.

10. `write_async(self: rti.connextdds.DynamicData.DataWriter, sample_data: dict) -> object`
Create a DynamicData object and write it with the given dictionary containing field names as keys. This method is awaitable and is only for use with asyncio.

class DataWriterListener

Bases: `pybind11_object`

`__init__` (*self*: `rti.connextdds.DynamicData.DataWriterListener`) → None

`__module__` = `'rti.connextdds'`

`on_application_acknowledgment` (*self*:
`rti.connextdds.DynamicData.DataWriterListener`,
arg0: `rti.connextdds.DynamicData.DataWriter`,
arg1: `rti.connextdds.AcknowledgmentInfo`) → None

On application acknowledgment callback

`on_instance_replaced` (*self*: `rti.connextdds.DynamicData.DataWriterListener`, *arg0*:
`rti.connextdds.DynamicData.DataWriter`, *arg1*:
`rti.connextdds.InstanceHandle`) → None

On instance replaced callback.

`on_liveliness_lost` (*self*: `rti.connextdds.DynamicData.DataWriterListener`, *arg0*:
`rti.connextdds.DynamicData.DataWriter`, *arg1*:
`rti.connextdds.LivelinessLostStatus`) → None

Liveliness lost callback.

`on_offered_deadline_missed` (*self*: `rti.connextdds.DynamicData.DataWriterListener`,
arg0: `rti.connextdds.DynamicData.DataWriter`, *arg1*:
`rti.connextdds.OfferedDeadlineMissedStatus`) → None

Offered deadline missed callback.

`on_offered_incompatible_qos` (*self*: `rti.connextdds.DynamicData.DataWriterListener`,
arg0: `rti.connextdds.DynamicData.DataWriter`, *arg1*:
`rti.connextdds.OfferedIncompatibleQosStatus`) → None

Offered incompatible QoS callback.

`on_publication_matched` (*self*: `rti.connextdds.DynamicData.DataWriterListener`, *arg0*:
`rti.connextdds.DynamicData.DataWriter`, *arg1*:
`rti.connextdds.PublicationMatchedStatus`) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connexdds.DynamicData.DataWriterListener, *arg0*: rti.connexdds.DynamicData.DataWriter, *arg1*: rti.connexdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connexdds.DynamicData.DataWriterListener, *arg0*: rti.connexdds.DynamicData.DataWriter, *arg1*: rti.connexdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connexdds.DynamicData.DataWriterListener, *arg0*: rti.connexdds.DynamicData.DataWriter, *arg1*: rti.connexdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class DataWriterSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.DynamicData.DataWriterSeq, *arg0*: rti.connexdds.DynamicData.DataWriterSeq) → *rti.connexdds.DynamicData.DataWriterSeq*

__bool__ (*self*: rti.connexdds.DynamicData.DataWriterSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.DynamicData.DataWriterSeq, *x*: rti.connexdds.DynamicData.DataWriter) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.DynamicData.DataWriterSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.DynamicData.DataWriterSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.DynamicData.DataWriterSeq, *arg0*: rti.connexdds.DynamicData.DataWriterSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.DynamicData.DataWriterSeq, *s*: slice) -> rti.connexdds.DynamicData.DataWriterSeq

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.DynamicData.DataWriterSeq, arg0: int) -> rti.connextdds.DynamicData.DataWriter`

`__hash__ = None`

`__iadd__` (*self: rti.connextdds.DynamicData.DataWriterSeq, arg0: rti.connextdds.DynamicData.DataWriterSeq*) → *rti.connextdds.DynamicData.DataWriterSeq*

`__imul__` (*self: rti.connextdds.DynamicData.DataWriterSeq, arg0: int*) → *rti.connextdds.DynamicData.DataWriterSeq*

`__init__` (**args, **kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.DynamicData.DataWriterSeq) -> None`
2. `__init__(self: rti.connextdds.DynamicData.DataWriterSeq, arg0: rti.connextdds.DynamicData.DataWriterSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.DynamicData.DataWriterSeq, arg0: Iterable) -> None`

`__iter__` (*self: rti.connextdds.DynamicData.DataWriterSeq*) → Iterator

`__len__` (*self: rti.connextdds.DynamicData.DataWriterSeq*) → int

`__module__ = 'rti.connextdds'`

`__mul__` (*self: rti.connextdds.DynamicData.DataWriterSeq, arg0: int*) → *rti.connextdds.DynamicData.DataWriterSeq*

`__ne__` (*self: rti.connextdds.DynamicData.DataWriterSeq, arg0: rti.connextdds.DynamicData.DataWriterSeq*) → bool

`__rmul__` (*self: rti.connextdds.DynamicData.DataWriterSeq, arg0: int*) → *rti.connextdds.DynamicData.DataWriterSeq*

`__setitem__` (**args, **kwargs*)

Overloaded function.

1. `__setitem__(self: rti.connextdds.DynamicData.DataWriterSeq, arg0: int, arg1: rti.connextdds.DynamicData.DataWriter) -> None`
2. `__setitem__(self: rti.connextdds.DynamicData.DataWriterSeq, arg0: slice, arg1: rti.connextdds.DynamicData.DataWriterSeq) -> None`

Assign list elements using a slice object

`append` (*self: rti.connextdds.DynamicData.DataWriterSeq, x: rti.connextdds.DynamicData.DataWriter*) → None

Add an item to the end of the list

`clear` (*self: rti.connextdds.DynamicData.DataWriterSeq*) → None

Clear the contents

count (*self*: rti.connextdds.DynamicData.DataWriterSeq, *x*: rti.connextdds.DynamicData.DataWriter) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connextdds.DynamicData.DataWriterSeq, *L*: rti.connextdds.DynamicData.DataWriterSeq) -> None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connextdds.DynamicData.DataWriterSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.DynamicData.DataWriterSeq, *i*: int, *x*: rti.connextdds.DynamicData.DataWriter) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connextdds.DynamicData.DataWriterSeq) -> rti.connextdds.DynamicData.DataWriter

Remove and return the last item

2. pop(*self*: rti.connextdds.DynamicData.DataWriterSeq, *i*: int) -> rti.connextdds.DynamicData.DataWriter

Remove and return the item at index *i*

remove (*self*: rti.connextdds.DynamicData.DataWriterSeq, *x*: rti.connextdds.DynamicData.DataWriter) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class FieldsIterator

Bases: pybind11_object

__init__ (*self*: rti.connextdds.DynamicData.FieldsIterator, *arg0*: rti.connextdds.DynamicData, *arg1*: bool) → None

__iter__ (*self*: rti.connextdds.DynamicData.FieldsIterator) → rti.connextdds.DynamicData.FieldsIterator

__module__ = 'rti.connextdds'

__next__ (*self*: rti.connextdds.DynamicData.FieldsIterator) → str

class FieldsView

Bases: pybind11_object

__contains__ (*self*: rti.connextdds.DynamicData.FieldsView, *arg0*: str) → bool

__init__ (*self*: rti.connextdds.DynamicData.FieldsView, *arg0*: rti.connextdds.DynamicData) → None

```

__iter__ (self: rti.connexdds.DynamicData.FieldsView) →
    rti.connexdds.DynamicData.FieldsIterator

__len__ (self: rti.connexdds.DynamicData.FieldsView) → int

__module__ = 'rti.connexdds'

__reversed__ (self: rti.connexdds.DynamicData.FieldsView) →
    rti.connexdds.DynamicData.FieldsIterator

```

class ITopicDescription

Bases: *IEntity*

```
__init__ (*args, **kwargs)
```

```
__module__ = 'rti.connexdds'
```

property name

The name of the entity conforming to the ITopicDescription interface.

property participant

The parent DomainParticipant.

property type_name

The name of the associated type.

class IndexIterator

Bases: *pybind11_object*

```
__init__ (self: rti.connexdds.DynamicData.IndexIterator, arg0:
    rti.connexdds.DynamicData, arg1: bool) → None
```

```
__iter__ (self: rti.connexdds.DynamicData.IndexIterator) →
    rti.connexdds.DynamicData.IndexIterator
```

```
__module__ = 'rti.connexdds'
```

```
__next__ (self: rti.connexdds.DynamicData.IndexIterator) → object
```

class ItemsIterator

Bases: *pybind11_object*

```
__init__ (self: rti.connexdds.DynamicData.ItemsIterator, arg0:
    rti.connexdds.DynamicData, arg1: bool) → None
```

```
__iter__ (self: rti.connexdds.DynamicData.ItemsIterator) →
    rti.connexdds.DynamicData.ItemsIterator
```

```
__module__ = 'rti.connexdds'
```

```
__next__ (self: rti.connexdds.DynamicData.ItemsIterator) → Tuple[str, object]
```

class ItemsView

Bases: pybind11_object

__contains__ (*self*: rti.connexdds.DynamicData.ItemsView, *arg0*: Tuple[str, object]) → bool**__init__** (*self*: rti.connexdds.DynamicData.ItemsView, *arg0*: rti.connexdds.DynamicData) → None**__iter__** (*self*: rti.connexdds.DynamicData.ItemsView) → *rti.connexdds.DynamicData.ItemsIterator***__len__** (*self*: rti.connexdds.DynamicData.ItemsView) → int**__module__** = 'rti.connexdds'**__reversed__** (*self*: rti.connexdds.DynamicData.ItemsView) → *rti.connexdds.DynamicData.ItemsIterator***class LoanedSample**

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DynamicData.LoanedSample) -> None

Basic constructor

2. **__init__**(*self*: rti.connexdds.DynamicData.LoanedSample, *data*: rti.connexdds.DynamicData, *info*: rti.connexdds.SampleInfo) -> None

Construct LoanedSample with data and info.

__iter__ (*self*: rti.connexdds.DynamicData.LoanedSample) → object**__module__** = 'rti.connexdds'**property data**

Get the data associated with the sample.

property info

Get the info associated with the sample.

class LoanedSamples

Bases: pybind11_object

__enter__ (*self*: rti.connexdds.DynamicData.LoanedSamples) → *rti.connexdds.DynamicData.LoanedSamples*

Enter a context for the loaned samples, loan returned on context exit.

__exit__ (*self*: rti.connexdds.DynamicData.LoanedSamples, *arg0*: object, *arg1*: object, *arg2*: object) → None

Exit the context for the loaned samples, returning the resources.

__getitem__ (*self*: rti.connextdds.DynamicData.LoanedSamples, *arg0*: int) → *rti.connextdds.DynamicData.LoanedSample*

Access a LoanedSample object in an array-like syntax

__init__ (*self*: rti.connextdds.DynamicData.LoanedSamples) → None

Create an empty LoanedSamples object.

__iter__ (*self*: rti.connextdds.DynamicData.LoanedSamples) → Iterator

__len__ (*self*: rti.connextdds.DynamicData.LoanedSamples) → int

Get the number of samples in the loan.

__module__ = 'rti.connextdds'

property length

Get the number of samples in the loan.

return_loan (*self*: rti.connextdds.DynamicData.LoanedSamples) → None

Returns the loan to the DataReader.

class NoOpDataReaderListener

Bases: *DataReaderListener*

__init__ (*self*: rti.connextdds.DynamicData.NoOpDataReaderListener) → None

__module__ = 'rti.connextdds'

on_data_available (*self*: rti.connextdds.DynamicData.NoOpDataReaderListener, *arg0*: *rti.connextdds.DynamicData.DataReader*) → None

Data available callback.

on_liveliness_changed (*self*: rti.connextdds.DynamicData.NoOpDataReaderListener, *arg0*: *rti.connextdds.DynamicData.DataReader*, *arg1*: *rti.connextdds.LivelinessChangedStatus*) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connextdds.DynamicData.NoOpDataReaderListener, *arg0*: *rti.connextdds.DynamicData.DataReader*, *arg1*: *rti.connextdds.RequestedDeadlineMissedStatus*) → None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connextdds.DynamicData.NoOpDataReaderListener, *arg0*: *rti.connextdds.DynamicData.DataReader*, *arg1*: *rti.connextdds.RequestedIncompatibleQosStatus*) → None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connextdds.DynamicData.NoOpDataReaderListener, *arg0*: rti.connextdds.DynamicData.DataReader, *arg1*: rti.connextdds.SampleLostStatus) → None

Sample lost callback.

on_sample_rejected (*self*: rti.connextdds.DynamicData.NoOpDataReaderListener, *arg0*: rti.connextdds.DynamicData.DataReader, *arg1*: rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connextdds.DynamicData.NoOpDataReaderListener, *arg0*: rti.connextdds.DynamicData.DataReader, *arg1*: rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class NoOpDataWriterListener

Bases: *DataWriterListener*

__init__ (*self*: rti.connextdds.DynamicData.NoOpDataWriterListener) → None

__module__ = 'rti.connextdds'

on_application_acknowledgment (*self*: rti.connextdds.DynamicData.NoOpDataWriterListener, *arg0*: rti.connextdds.DynamicData.DataWriter, *arg1*: rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

on_instance_replaced (*self*: rti.connextdds.DynamicData.NoOpDataWriterListener, *arg0*: rti.connextdds.DynamicData.DataWriter, *arg1*: rti.connextdds.InstanceHandle) → None

On instance replaced callback.

on_liveliness_lost (*self*: rti.connextdds.DynamicData.NoOpDataWriterListener, *arg0*: rti.connextdds.DynamicData.DataWriter, *arg1*: rti.connextdds.LivelinessLostStatus) → None

Liveliness lost callback.

on_offered_deadline_missed (*self*: rti.connextdds.DynamicData.NoOpDataWriterListener, *arg0*: rti.connextdds.DynamicData.DataWriter, *arg1*: rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

on_offered_incompatible_qos (*self*: rti.connextdds.DynamicData.NoOpDataWriterListener, *arg0*: rti.connextdds.DynamicData.DataWriter, *arg1*: rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

```
on_publication_matched (self: rti.connexdds.DynamicData.NoOpDataWriterListener,
                        arg0: rti.connexdds.DynamicData.DataWriter, arg1:
                        rti.connexdds.PublicationMatchedStatus) → None
```

Publication matched callback.

```
on_reliable_reader_activity_changed (self: rti.connexdds.DynamicData.NoOp-
                                      DataWriterListener, arg0:
                                      rti.connexdds.DynamicData.DataWriter,
                                      arg1: rti.connexdds.ReliableReaderAc-
                                      tivityChangedStatus) →
                                      None
```

Reliable reader activity changed callback.

```
on_reliable_writer_cache_changed (self: rti.connexdds.DynamicData.NoOp-
                                      DataWriterListener, arg0:
                                      rti.connexdds.DynamicData.DataWriter,
                                      arg1: rti.connexdds.ReliableWriter-
                                      CacheChangedStatus) →
                                      None
```

Reliable writer cache changed callback.

```
on_service_request_accepted (self: rti.connexdds.DynamicData.NoOp-
                                 DataWriterListener, arg0:
                                 rti.connexdds.DynamicData.DataWriter, arg1:
                                 rti.connexdds.ServiceRequestAcceptedStatus) →
                                 None
```

On service request accepted callback.

```
class NoOpTopicListener
```

Bases: *TopicListener*

```
__init__ (self: rti.connexdds.DynamicData.NoOpTopicListener) → None
```

```
__module__ = 'rti.connexdds'
```

```
on_inconsistent_topic (self: rti.connexdds.DynamicData.NoOpTopicListener, arg0:
                        rti.connexdds.DynamicData.Topic, arg1:
                        rti.connexdds.InconsistentTopicStatus) → None
```

Inconsistent topic callback.

```
class Sample
```

Bases: *pybind11_object*

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connexdds.DynamicData.Sample, data: rti.connexdds.DynamicData, info: rti.connexdds.SampleInfo) -> None`

Construct Sample with data and info.

2. `__init__(self: rti.connextdds.DynamicData.Sample, sample: rti.connextdds.DynamicData.Sample) -> None`

Copy constructor.

3. `__init__(self: rti.connextdds.DynamicData.Sample, loaned_sample: rti.connextdds.DynamicData.LoanedSample) -> None`

Copy constructor.

`__iter__(self: rti.connextdds.DynamicData.Sample) → object`

`__module__ = 'rti.connextdds'`

property data

The data associated with the sample.

property info

Get the info associated with the sample.

class SharedSamples

Bases: `pybind11_object`

`__getitem__(self: rti.connextdds.DynamicData.SharedSamples, arg0: int) → rti.connextdds.DynamicData.LoanedSample`

Get the sample at the specified index.

`__init__(self: rti.connextdds.DynamicData.SharedSamples, loaned_samples: rti.connextdds.DynamicData.LoanedSamples) → None`

Constructs an instance of SharedSamples, removing ownership of the loan from the Loaned Samples.

`__iter__(self: rti.connextdds.DynamicData.SharedSamples) → Iterator`

Make a sample iterator

`__len__(self: rti.connextdds.DynamicData.SharedSamples) → int`

Returns the number of samples.

`__module__ = 'rti.connextdds'`

class Topic

Bases: `ITopicDescription, IAnyTopic`

`__eq__(self: rti.connextdds.DynamicData.Topic, arg0: rti.connextdds.DynamicData.Topic) → bool`

Test for equality.

`__hash__ = None`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.DynamicData.Topic, entity: rti.connextdds.IEntity) -> None`

Cast an Entity to a Topic.

2. `__init__(self: rti.connextdds.DynamicData.Topic, topic_description: rti.connextdds.DynamicData.ITopicDescription) -> None`

Cast an ITopicDescription to a Topic.

3. `__init__(self: rti.connextdds.DynamicData.Topic, topic: rti.connextdds.IAnyTopic) -> None`

Create a typed Topic from an AnyTopic.

4. `__init__(self: rti.connextdds.DynamicData.Topic, participant: rti.connextdds.DomainParticipant, topic_name: str, topic_type: rti.connextdds.DynamicType) -> None`

Create a Topic with the given type.

5. `__init__(self: rti.connextdds.DynamicData.Topic, participant: rti.connextdds.DomainParticipant, topic_name: str, topic_type: rti.connextdds.DynamicType, qos: rti.connextdds.TopicQos, listener: Optional[rti.connextdds.DynamicData.TopicListener] = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Create a Topic with the given type.

6. `__init__(self: rti.connextdds.DynamicData.Topic, participant: rti.connextdds.DomainParticipant, topic_name: str, type_name: str, qos: rti.connextdds.TopicQos, listener: Optional[rti.connextdds.DynamicData.TopicListener] = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Creates a new Topic.

`__module__ = 'rti.connextdds'`

`__ne__` (*self*: rti.connextdds.DynamicData.Topic, *arg0*: rti.connextdds.DynamicData.Topic) → bool

Test for inequality.

`static find` (*participant*: rti.connextdds.DomainParticipant, *name*: str) → Optional[rti.connextdds.DynamicData.Topic]

Look up a Topic by its name in the DomainParticipant.

`property inconsistent_topic_status`

Get a copy of the current InconsistentTopicStatus for this Topic.

`property listener`

The listener.

`property qos`

Get the TopicQos for this Topic.

This property's getter returns a deep copy.

`set_listener` (*self*: rti.connextdds.DynamicData.Topic, *listener*: rti.connextdds.DynamicData.TopicListener, *event_mask*: rti.connextdds.StatusMask) → None

Set the listener and event mask.

`class TopicDescription`

Bases: *ITopicDescription*

__eq__ (*self*: rti.connexdds.DynamicData.TopicDescription, *arg0*: rti.connexdds.DynamicData.TopicDescription) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DynamicData.TopicDescription, *topic*: rti.connexdds.DynamicData.ITopicDescription) -> None

Cast a Topic to a TopicDescription.

2. **__init__**(*self*: rti.connexdds.DynamicData.TopicDescription, *entity*: rti.connexdds.IEntity) -> None

Cast a Topic to a TopicDescription.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DynamicData.TopicDescription, *arg0*: rti.connexdds.DynamicData.TopicDescription) → bool

Test for inequality.

class TopicListener

Bases: pybind11_object

__init__ (*self*: rti.connexdds.DynamicData.TopicListener) → None

__module__ = 'rti.connexdds'

on_inconsistent_topic (*self*: rti.connexdds.DynamicData.TopicListener, *arg0*: rti.connexdds.DynamicData.Topic, *arg1*: rti.connexdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class TopicSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: rti.connexdds.DynamicData.TopicSeq) → *rti.connexdds.DynamicData.TopicSeq*

__bool__ (*self*: rti.connexdds.DynamicData.TopicSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.DynamicData.TopicSeq, *x*: rti.connexdds.DynamicData.Topic) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: rti.connexdds.DynamicData.TopicSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.DynamicData.TopicSeq, *s*: slice) -> rti.connexdds.DynamicData.TopicSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: int) -> rti.connexdds.DynamicData.Topic

__hash__ = None

__iadd__ (*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: rti.connexdds.DynamicData.TopicSeq) → *rti.connexdds.DynamicData.TopicSeq*

__imul__ (*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: int) → *rti.connexdds.DynamicData.TopicSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DynamicData.TopicSeq) -> None

2. **__init__**(*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: rti.connexdds.DynamicData.TopicSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.DynamicData.TopicSeq) → Iterator

__len__ (*self*: rti.connexdds.DynamicData.TopicSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: int) → *rti.connexdds.DynamicData.TopicSeq*

__ne__ (*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: rti.connexdds.DynamicData.TopicSeq) → bool

__rmul__ (*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: int) → *rti.connexdds.DynamicData.TopicSeq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: int, *arg1*: rti.connexdds.DynamicData.Topic) -> None

2. **__setitem__**(*self*: rti.connexdds.DynamicData.TopicSeq, *arg0*: slice, *arg1*: rti.connexdds.DynamicData.TopicSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.DynamicData.TopicSeq, *x*: rti.connexdds.DynamicData.Topic) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.DynamicData.TopicSeq) → None

Clear the contents

count (*self*: rti.connexdds.DynamicData.TopicSeq, *x*: rti.connexdds.DynamicData.Topic) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. `extend(self: rti.connexdds.DynamicData.TopicSeq, L: rti.connexdds.DynamicData.TopicSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.DynamicData.TopicSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.DynamicData.TopicSeq, *i*: int, *x*: rti.connexdds.DynamicData.Topic) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connexdds.DynamicData.TopicSeq) -> rti.connexdds.DynamicData.Topic`
Remove and return the last item

2. `pop(self: rti.connexdds.DynamicData.TopicSeq, i: int) -> rti.connexdds.DynamicData.Topic`

Remove and return the item at index *i*

remove (*self*: rti.connexdds.DynamicData.TopicSeq, *x*: rti.connexdds.DynamicData.Topic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class TopicTypeSupport

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

static from_cdr_buffer (*sample*: rti.connexdds.DynamicData, *buffer*: rti.connexdds.CharSeq) → None

Deserialize a sample from a CDR buffer.

static initialize_sample (*arg0*: rti.connexdds.DynamicData) → None

static register_type (*participant*: rti.connexdds.DomainParticipant, *type_name*: str) → None

Register a type with a participant.

```
static to_cdr_buffer (buffer: rti.connextdds.CharSeq, sample:
    rti.connextdds.DynamicData, representation_id: int = -1) →
    rti.connextdds.CharSeq
```

Serialize sample to a CDR buffer.

class ValidLoanedSamples

Bases: *pybind11_object*

```
__enter__ (self: rti.connextdds.DynamicData.ValidLoanedSamples) →
    rti.connextdds.DynamicData.ValidLoanedSamples
```

```
__exit__ (self: rti.connextdds.DynamicData.ValidLoanedSamples, arg0: object, arg1: object,
    arg2: object) → None
```

```
__init__ (*args, **kwargs)
```

```
__iter__ (self: rti.connextdds.DynamicData.ValidLoanedSamples) → Iterator
```

```
__module__ = 'rti.connextdds'
```

class WriterContentFilter

Bases: *ContentFilter*

```
__init__ (self: rti.connextdds.DynamicData.WriterContentFilter) → None
```

```
__module__ = 'rti.connextdds'
```

```
writer_attach (self: rti.connextdds.DynamicData.WriterContentFilter) → Optional[object]
```

A writer-side filtering API to create some state that can facilitate filtering on the writer side.

```
writer_compile (self: rti.connextdds.DynamicData.WriterContentFilter, writer_filter_data:
    Optional[object], property: rti.connextdds.ExpressionProperty, expression:
    str, parameters: rti.connextdds.StringSeq, type_code:
    Optional[rti.connextdds.DynamicType], type_class_name: str, cookie:
    rti.connextdds.Cookie) → None
```

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

```
writer_detach (self: rti.connextdds.DynamicData.WriterContentFilter, writer_filter_data:
    Optional[object]) → None
```

A writer-side filtering API to clean up a previously created state using `writer_attach`.

```
writer_evaluate (self: rti.connextdds.DynamicData.WriterContentFilter,
    writer_filter_data: Optional[object], sample:
    rti.connextdds.DynamicData, meta_data:
    rti.connextdds.FilterSampleInfo) → rti.connextdds.CookieVector
```

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

writer_finalize (*self*: rti.connexdds.DynamicData.WriterContentFilter, *writer_filter_data*: *Optional*[*object*], *cookie*: rti.connexdds.Cookie) → None

A writer-side filtering API to clean up a previously compiled instance of the content filter.

writer_return_loan (*self*: rti.connexdds.DynamicData.WriterContentFilter, *writer_filter_data*: *Optional*[*object*], *cookies*: rti.connexdds.CookieVector) → None

A writer-side filtering API to return the loan on the list of DataReaders returned by `writer_evaluate`.

class WriterContentFilterHelper

Bases: *WriterContentFilter*

__init__ (*self*: rti.connexdds.DynamicData.WriterContentFilterHelper) → None

__module__ = 'rti.connexdds'

add_cookie (*self*: rti.connexdds.DynamicData.WriterContentFilterHelper, *cookie*: rti.connexdds.Cookie) → None

A helper function which will add a Cookie to the Cookie sequence that is then returned by the `writer_evaluate` function.

writer_evaluate_helper (*self*: rti.connexdds.DynamicData.WriterContentFilterHelper, *writer_filter_data*: *Optional*[*object*], *sample*: rti.connexdds.DynamicData, *meta_data*: rti.connexdds.FilterSampleInfo) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

__contains__ (**args*, ***kwargs*)

Overloaded function.

1. `__contains__(self: rti.connexdds.DynamicData, arg0: str) -> bool`
2. `__contains__(self: rti.connexdds.DynamicData, arg0: object) -> bool`

__eq__ (**args*, ***kwargs*)

Overloaded function.

1. `__eq__(self: rti.connexdds.DynamicData, arg0: rti.connexdds.DynamicData) -> bool`

Test for equality.

2. `__eq__(self: rti.connexdds.DynamicData, arg0: dict) -> bool`

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__(self: rti.connexdds.DynamicData, arg0: str) -> object`
2. `__getitem__(self: rti.connexdds.DynamicData, arg0: int) -> object`
3. `__getitem__(self: rti.connexdds.DynamicData, arg0: tuple) -> object`

4. `__getitem__(self: rti.connextdds.DynamicData, arg0: slice) -> list`

`__hash__ = None`

`__init__ (*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.DynamicData, dynamic_type: rti.connextdds.DynamicType) -> None`

Create a DynamicData object from a DynamicType.

2. `__init__(self: rti.connextdds.DynamicData, dynamic_type: rti.connextdds.DynamicType, property: rti.connextdds.DynamicDataProperty) -> None`

Create a DynamicData object from a DynamicType with properties.

3. `__init__(self: rti.connextdds.DynamicData, dynamic_type: rti.connextdds.DynamicType, data: dict) -> None`

Create a DynamicData object from a DynamicType with a dict to initialize the fields

4. `__init__(self: rti.connextdds.DynamicData, other: rti.connextdds.DynamicData) -> None`

Create a copy of a DynamicData object.

`__iter__(self: rti.connextdds.DynamicData) → object`

`__len__(self: rti.connextdds.DynamicData) → int`

Number of members in the DynamicData object

`__module__ = 'rti.connextdds'`

`__ne__ (*args, **kwargs)`

Overloaded function.

1. `__ne__(self: rti.connextdds.DynamicData, arg0: rti.connextdds.DynamicData) -> bool`

Test for inequality.

2. `__ne__(self: rti.connextdds.DynamicData, arg0: dict) -> bool`

`__reversed__(self: rti.connextdds.DynamicData) → object`

`__setitem__ (*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connextdds.DynamicData, arg0: str, arg1: object) -> None`

2. `__setitem__(self: rti.connextdds.DynamicData, arg0: int, arg1: object) -> None`

3. `__setitem__(self: rti.connextdds.DynamicData, arg0: tuple, arg1: object) -> None`

4. `__setitem__(self: rti.connextdds.DynamicData, arg0: slice, arg1: Iterable) -> None`

`__str__(self: rti.connextdds.DynamicData) → str`

DynamicData value to string.

append (*self*: rti.connexdds.DynamicData, *value*: *object*) → None

Append a value to a sequence.

clear_all_members (*self*: rti.connexdds.DynamicData) → None

Clear the contents of all data members of this object, including key members.

clear_member (**args*, ***kwargs*)

Overloaded function.

1. `clear_member(self: rti.connexdds.DynamicData, name: str) -> None`

Clear the contents of a single data member of this object.

2. `clear_member(self: rti.connexdds.DynamicData, index: int) -> None`

Clear the contents of a single data member of this object.

clear_optional_member (**args*, ***kwargs*)

Overloaded function.

1. `clear_optional_member(self: rti.connexdds.DynamicData, name: str) -> None`

Clear the contents of a single optional data member of this object.

2. `clear_optional_member(self: rti.connexdds.DynamicData, index: int) -> None`

Clear the contents of a single optional data member of this object.

count (*self*: rti.connexdds.DynamicData, *arg0*: *object*) → int

property discriminator_value

Obtains the value of the union discriminator (valid for UnionType only).

extend (*self*: rti.connexdds.DynamicData, *values*: *Iterable*) → None

Extend a sequence.

fields (*self*: rti.connexdds.DynamicData) → *rti.connexdds.DynamicData.FieldsView*

from_cdr_buffer (**args*, ***kwargs*)

Overloaded function.

1. `from_cdr_buffer(self: rti.connexdds.DynamicData, buffer: rti.connexdds.Uint8Seq) -> rti.connexdds.DynamicData`

Populates a DynamicData sample by deserializing a CDR buffer.

2. `from_cdr_buffer(self: rti.connexdds.DynamicData, signed_buffer: rti.connexdds.CharSeq) -> rti.connexdds.DynamicData`

Populates a DynamicData sample by deserializing a CDR buffer.

from_json (*self*: rti.connexdds.DynamicData, *str*: *str*) → None

Populates a DynamicData object from a JSON string representation.

from_string (*self*: rti.connextdds.DynamicData, *str*: str, *format_kind*: rti.connextdds.PrintFormatKind) → None

Populates a DynamicData object from a JSON string representation. At the moment the only supported value for *format_kind* is `dds.PrintFormatKind.JSON`, and `from_json()` is preferred.

get_boolean (**args*, ***kwargs*)

Overloaded function.

1. `get_boolean(self: rti.connextdds.DynamicData, name: str) -> bool`

Get a boolean value by field name.

2. `get_boolean(self: rti.connextdds.DynamicData, name: int) -> bool`

Get a boolean value by field name.

get_cdr_buffer (*self*: rti.connextdds.DynamicData) → memoryview

Gets the CDR buffer of this DynamicData sample as a memoryview.

get_char (**args*, ***kwargs*)

Overloaded function.

1. `get_char(self: rti.connextdds.DynamicData, name: str) -> str`

Get a character value by field name.

2. `get_char(self: rti.connextdds.DynamicData, name: int) -> str`

Get a character value by field name.

get_char_values (**args*, ***kwargs*)

Overloaded function.

1. `get_char_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.CharSeq`

Get multiple character values by field name.

2. `get_char_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.CharSeq`

Get multiple character values by field name.

get_complex (**args*, ***kwargs*)

Overloaded function.

1. `get_complex(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.DynamicData`

Get a complex data type value by field name.

2. `get_complex(self: rti.connextdds.DynamicData, name: int) -> rti.connextdds.DynamicData`

Get a complex data type value by field name.

get_complex_values (**args*, ***kwargs*)

Overloaded function.

1. `get_complex_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.DynamicDataSeq`

Get a list of complex values by field name.

2. `get_complex_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.DynamicDataSeq`

Get a list of complex fields by index.

get_double (*args, **kwargs)

Overloaded function.

1. `get_double(self: rti.connextdds.DynamicData, name: str) -> float`

Get a 64-bit floating point value by field name.

2. `get_double(self: rti.connextdds.DynamicData, name: int) -> float`

Get a 64-bit floating point value by field name.

get_double_values (*args, **kwargs)

Overloaded function.

1. `get_double_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Float64Seq`

Get multiple 64-bit floating point values by field name.

2. `get_double_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Float64Seq`

Get multiple 64-bit floating point values by field name.

get_float (*args, **kwargs)

Overloaded function.

1. `get_float(self: rti.connextdds.DynamicData, name: str) -> float`

Get a 32-bit floating point value by field name.

2. `get_float(self: rti.connextdds.DynamicData, name: int) -> float`

Get a 32-bit floating point value by field name.

get_float128 (*args, **kwargs)

Overloaded function.

1. `get_float128(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.LongDouble`

Get a 128-bit floating point value by field name.

2. `get_float128(self: rti.connextdds.DynamicData, name: int) -> rti.connextdds.LongDouble`

Get a 128-bit floating point value by field name.

get_float32 (*args, **kwargs)

Overloaded function.

1. `get_float32(self: rti.connextdds.DynamicData, name: str) -> float`

Get a 32-bit floating point value by field name.

2. `get_float32(self: rti.connextdds.DynamicData, name: int) -> float`

Get a 32-bit floating point value by field name.

get_float32_values (*args, **kwargs)

Overloaded function.

1. `get_float32_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Float32Seq`

Get multiple 32-bit floating point values by field name.

2. `get_float32_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Float32Seq`

Get multiple 32-bit floating point values by field name.

get_float64 (*args, **kwargs)

Overloaded function.

1. `get_float64(self: rti.connextdds.DynamicData, name: str) -> float`

Get a 64-bit floating point value by field name.

2. `get_float64(self: rti.connextdds.DynamicData, name: int) -> float`

Get a 64-bit floating point value by field name.

get_float64_values (*args, **kwargs)

Overloaded function.

1. `get_float64_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Float64Seq`

Get multiple 64-bit floating point values by field name.

2. `get_float64_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Float64Seq`

Get multiple 64-bit floating point values by field name.

get_float_values (*args, **kwargs)

Overloaded function.

1. `get_float_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Float32Seq`

Get multiple 32-bit floating point values by field name.

2. `get_float_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Float32Seq`

Get multiple 32-bit floating point values by field name.

get_int (*args, **kwargs)

Overloaded function.

1. `get_int(self: rti.connextdds.DynamicData, name: str) -> int`

Get a int (signed) value by field name.

2. `get_int(self: rti.connextdds.DynamicData, name: int) -> int`

Get a int (signed) value by field name.

`get_int16 (*args, **kwargs)`

Overloaded function.

1. `get_int16(self: rti.connextdds.DynamicData, name: str) -> int`

Get a 16-bit signed int value by field name.

2. `get_int16(self: rti.connextdds.DynamicData, name: int) -> int`

Get a 16-bit signed int value by field name.

`get_int16_values (*args, **kwargs)`

Overloaded function.

1. `get_int16_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Int16Seq`

Get multiple 16-bit signed int values by field name.

2. `get_int16_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Int16Seq`

Get multiple 16-bit signed int values by field name.

`get_int32 (*args, **kwargs)`

Overloaded function.

1. `get_int32(self: rti.connextdds.DynamicData, name: str) -> int`

Get a 32-bit signed int value by field name.

2. `get_int32(self: rti.connextdds.DynamicData, name: int) -> int`

Get a 32-bit signed int value by field name.

`get_int32_values (*args, **kwargs)`

Overloaded function.

1. `get_int32_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Int32Seq`

Get multiple 32-bit signed int values by field name.

2. `get_int32_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Int32Seq`

Get multiple 32-bit signed int values by field name.

`get_int64 (*args, **kwargs)`

Overloaded function.

1. `get_int64(self: rti.connextdds.DynamicData, name: str) -> int`

Get a 64-bit signed int value by field name.

2. `get_int64(self: rti.connextdds.DynamicData, name: int) -> int`

Get a 64-bit signed int value by field name.

get_int64_values (*args, **kwargs)

Overloaded function.

1. get_int64_values(self: rti.connextdds.DynamicData, name: str) -> List[int]

Get multiple 64-bit signed int values by field name.

2. get_int64_values(self: rti.connextdds.DynamicData, index: int) -> List[int]

Get multiple 64-bit signed int values by field name.

get_int8 (*args, **kwargs)

Overloaded function.

1. get_int8(self: rti.connextdds.DynamicData, name: str) -> int

Get an int8 value by field name.

2. get_int8(self: rti.connextdds.DynamicData, index: int) -> int

Get an int8 value by field index.

get_int_values (*args, **kwargs)

Overloaded function.

1. get_int_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Int32Seq

Get multiple 32-bit signed int values by field name.

2. get_int_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Int32Seq

Get multiple 32-bit signed int values by field name.

get_long (*args, **kwargs)

Overloaded function.

1. get_long(self: rti.connextdds.DynamicData, name: str) -> int

Get a 32-bit signed int value by field name.

2. get_long(self: rti.connextdds.DynamicData, name: int) -> int

Get a 32-bit signed int value by field name.

get_long_values (*args, **kwargs)

Overloaded function.

1. get_long_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Int32Seq

Get multiple 32-bit signed int values by field name.

2. get_long_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Int32Seq

Get multiple 32-bit signed int values by field name.

get_longdouble (*args, **kwargs)

Overloaded function.

1. get_longdouble(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.LongDouble

Get a 128-bit floating point value by field name.

2. `get_longdouble(self: rti.connextdds.DynamicData, name: int) -> rti.connextdds.LongDouble`

Get a 128-bit floating point value by field name.

get_longlong (*args, **kwargs)

Overloaded function.

1. `get_longlong(self: rti.connextdds.DynamicData, name: str) -> int`

Get a 64-bit signed int value by field name.

2. `get_longlong(self: rti.connextdds.DynamicData, name: int) -> int`

Get a 64-bit signed int value by field name.

get_longlong_values (*args, **kwargs)

Overloaded function.

1. `get_longlong_values(self: rti.connextdds.DynamicData, name: str) -> List[int]`

Get multiple 64-bit signed int values by field name.

2. `get_longlong_values(self: rti.connextdds.DynamicData, index: int) -> List[int]`

Get multiple 64-bit signed int values by field name.

get_octet (*args, **kwargs)

Overloaded function.

1. `get_octet(self: rti.connextdds.DynamicData, name: str) -> int`

Get a 8-bit unsigned int value by field name.

2. `get_octet(self: rti.connextdds.DynamicData, name: int) -> int`

Get a 8-bit unsigned int value by field name.

get_octet_values (*args, **kwargs)

Overloaded function.

1. `get_octet_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Uint8Seq`

Get multiple 8-bit unsigned int values by field name.

2. `get_octet_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Uint8Seq`

Get multiple 8-bit unsigned int values by field name.

get_short (*args, **kwargs)

Overloaded function.

1. `get_short(self: rti.connextdds.DynamicData, name: str) -> int`

Get a 16-bit signed int value by field name.

2. `get_short(self: rti.connextdds.DynamicData, name: int) -> int`

Get a 16-bit signed int value by field name.

get_short_values (*args, **kwargs)

Overloaded function.

1. `get_short_values(self: rti.connexdds.DynamicData, name: str) -> rti.connexdds.Int16Seq`

Get multiple 16-bit signed int values by field name.

2. `get_short_values(self: rti.connexdds.DynamicData, index: int) -> rti.connexdds.Int16Seq`

Get multiple 16-bit signed int values by field name.

get_string (*args, **kwargs)

Overloaded function.

1. `get_string(self: rti.connexdds.DynamicData, name: str) -> str`

Get a string value by field name.

2. `get_string(self: rti.connexdds.DynamicData, name: int) -> str`

Get a string value by field name.

get_uint (*args, **kwargs)

Overloaded function.

1. `get_uint(self: rti.connexdds.DynamicData, name: str) -> int`

Get a 32-bit unsigned int value by field name.

2. `get_uint(self: rti.connexdds.DynamicData, name: int) -> int`

Get a 32-bit unsigned int value by field name.

get_uint16 (*args, **kwargs)

Overloaded function.

1. `get_uint16(self: rti.connexdds.DynamicData, name: str) -> int`

Get a 16-bit unsigned int value by field name.

2. `get_uint16(self: rti.connexdds.DynamicData, name: int) -> int`

Get a 16-bit unsigned int value by field name.

get_uint16_values (*args, **kwargs)

Overloaded function.

1. `get_uint16_values(self: rti.connexdds.DynamicData, name: str) -> rti.connexdds.Uint16Seq`

Get multiple 16-bit unsigned int values by field name.

2. `get_uint16_values(self: rti.connexdds.DynamicData, index: int) -> rti.connexdds.Uint16Seq`

Get multiple 16-bit unsigned int values by field name.

get_uint32 (*args, **kwargs)

Overloaded function.

1. get_uint32(self: rti.connextdds.DynamicData, name: str) -> int

Get a 32-bit unsigned int value by field name.

2. get_uint32(self: rti.connextdds.DynamicData, name: int) -> int

Get a 32-bit unsigned int value by field name.

get_uint32_values (*args, **kwargs)

Overloaded function.

1. get_uint32_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Uint32Seq

Get multiple 32-bit unsigned int values by field name.

2. get_uint32_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Uint32Seq

Get multiple 32-bit unsigned int values by field name.

get_uint64 (*args, **kwargs)

Overloaded function.

1. get_uint64(self: rti.connextdds.DynamicData, name: str) -> int

Get a 64-bit unsigned int value by field name.

2. get_uint64(self: rti.connextdds.DynamicData, name: int) -> int

Get a 64-bit unsigned int value by field name.

get_uint64_values (*args, **kwargs)

Overloaded function.

1. get_uint64_values(self: rti.connextdds.DynamicData, name: str) -> List[int]

Get multiple 64-bit unsigned int values by field name.

2. get_uint64_values(self: rti.connextdds.DynamicData, index: int) -> List[int]

Get multiple 64-bit unsigned int values by field name.

get_uint8 (*args, **kwargs)

Overloaded function.

1. get_uint8(self: rti.connextdds.DynamicData, name: str) -> int

Get a 8-bit unsigned int value by field name.

2. get_uint8(self: rti.connextdds.DynamicData, name: int) -> int

Get a 8-bit unsigned int value by field name.

get_uint8_values (*args, **kwargs)

Overloaded function.

1. `get_uint8_values(self: rti.connexdds.DynamicData, name: str) -> rti.connexdds.Uint8Seq`
Get multiple 8-bit unsigned int values by field name.
2. `get_uint8_values(self: rti.connexdds.DynamicData, index: int) -> rti.connexdds.Uint8Seq`
Get multiple 8-bit unsigned int values by field name.

get_uint_values (*args, **kwargs)

Overloaded function.

1. `get_uint_values(self: rti.connexdds.DynamicData, name: str) -> rti.connexdds.Uint32Seq`
Get multiple 32-bit unsigned int values by field name.
2. `get_uint_values(self: rti.connexdds.DynamicData, index: int) -> rti.connexdds.Uint32Seq`
Get multiple 32-bit unsigned int values by field name.

get_ulong (*args, **kwargs)

Overloaded function.

1. `get_ulong(self: rti.connexdds.DynamicData, name: str) -> int`
Get a 32-bit unsigned int value by field name.
2. `get_ulong(self: rti.connexdds.DynamicData, name: int) -> int`
Get a 32-bit unsigned int value by field name.

get_ulong_values (*args, **kwargs)

Overloaded function.

1. `get_ulong_values(self: rti.connexdds.DynamicData, name: str) -> rti.connexdds.Uint32Seq`
Get multiple 32-bit unsigned int values by field name.
2. `get_ulong_values(self: rti.connexdds.DynamicData, index: int) -> rti.connexdds.Uint32Seq`
Get multiple 32-bit unsigned int values by field name.

get_ulonglong (*args, **kwargs)

Overloaded function.

1. `get_ulonglong(self: rti.connexdds.DynamicData, name: str) -> int`
Get a 64-bit unsigned int value by field name.
2. `get_ulonglong(self: rti.connexdds.DynamicData, name: int) -> int`
Get a 64-bit unsigned int value by field name.

get_ulonglong_values (*args, **kwargs)

Overloaded function.

1. `get_ulonglong_values(self: rti.connexdds.DynamicData, name: str) -> List[int]`

Get multiple 64-bit unsigned int values by field name.

2. `get_ulonglong_values(self: rti.connextdds.DynamicData, index: int) -> List[int]`

Get multiple 64-bit unsigned int values by field name.

get_ushort (*args, **kwargs)

Overloaded function.

1. `get_ushort(self: rti.connextdds.DynamicData, name: str) -> int`

Get a 16-bit unsigned int value by field name.

2. `get_ushort(self: rti.connextdds.DynamicData, name: int) -> int`

Get a 16-bit unsigned int value by field name.

get_ushort_values (*args, **kwargs)

Overloaded function.

1. `get_ushort_values(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.Uint16Seq`

Get multiple 16-bit unsigned int values by field name.

2. `get_ushort_values(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.Uint16Seq`

Get multiple 16-bit unsigned int values by field name.

get_value (*args, **kwargs)

Overloaded function.

1. `get_value(self: rti.connextdds.DynamicData, field_path: str) -> object`

Automatically resolve type and return value for a field.

2. `get_value(self: rti.connextdds.DynamicData, field_path: int) -> object`

Automatically resolve type and return value for a field.

get_values (*args, **kwargs)

Overloaded function.

1. `get_values(self: rti.connextdds.DynamicData, field_path: str) -> object`

Automatically resolve type and return collection for a field.

2. `get_values(self: rti.connextdds.DynamicData, field_path: int) -> object`

Automatically resolve type and return collection for a field.

get_wchar (*args, **kwargs)

Overloaded function.

1. `get_wchar(self: rti.connextdds.DynamicData, name: str) -> str`

Get a wchar value by field name.

2. `get_wchar(self: rti.connextdds.DynamicData, index: int) -> str`

Get a wchar value by field index.

get_wstring (*args, **kwargs)

Overloaded function.

1. get_wstring(self: rti.connextdds.DynamicData, name: str) -> str

Get a wstring value by field name.

2. get_wstring(self: rti.connextdds.DynamicData, index: int) -> str

Get a wstring value by field index.

property info

Returns info about this sample

property is_cdr

Determines if this DynamicData sample is represented as CDR

is_member_key (*args, **kwargs)

Overloaded function.

1. is_member_key(self: rti.connextdds.DynamicData, name: str) -> bool

Returns whether a member is a key.

2. is_member_key(self: rti.connextdds.DynamicData, index: int) -> bool

Returns whether a member is a key.

items (self: rti.connextdds.DynamicData) → *rti.connextdds.DynamicData.ItemsView*

loan_value (*args, **kwargs)

Overloaded function.

1. loan_value(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.LoanedDynamicData

Gets a view of a complex member.

2. loan_value(self: rti.connextdds.DynamicData, data: rti.connextdds.LoanedDynamicData, index: int) -> rti.connextdds.LoanedDynamicData

Gets a view of a complex member.

3. loan_value(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.LoanedDynamicData

Gets a view of a complex member.

4. loan_value(self: rti.connextdds.DynamicData, data: rti.connextdds.LoanedDynamicData, name: str) -> rti.connextdds.LoanedDynamicData

Gets a view of a complex member.

property member_count

Get the number of members in this sample.

member_exists (*args, **kwargs)

Overloaded function.

1. member_exists(self: rti.connextdds.DynamicData, name: str) -> bool

Determine if an optional member is set by member name.

2. member_exists(self: rti.connextdds.DynamicData, index: int) -> bool

Determine if an optional member is set by member index.

member_exists_in_type (*args, **kwargs)

Overloaded function.

1. member_exists_in_type(self: rti.connextdds.DynamicData, name: str) -> bool

Determine if a member with a particular name exists in the type.

2. member_exists_in_type(self: rti.connextdds.DynamicData, index: int) -> bool

Determine if a member with a particular index exists in the type.

member_index (self: rti.connextdds.DynamicData, name: str) → int

Translates from member name to member index.

member_info (*args, **kwargs)

Overloaded function.

1. member_info(self: rti.connextdds.DynamicData, name: str) -> rti.connextdds.DynamicDataMemberInfo

Returns info about a member.

2. member_info(self: rti.connextdds.DynamicData, index: int) -> rti.connextdds.DynamicDataMemberInfo

Returns info about a member.

set_boolean (*args, **kwargs)

Overloaded function.

1. set_boolean(self: rti.connextdds.DynamicData, name: str, value: bool) -> None

Set a boolean value by name.

2. set_boolean(self: rti.connextdds.DynamicData, index: int, value: bool) -> None

Set a boolean value by index.

set_cdr_buffer (self: rti.connextdds.DynamicData, buffer: buffer) → None

Sets the CDR buffer of this DynamicData sample.

buffer can be any object that supports the buffer protocol and produces a one-dimensional contiguous byte buffer.

set_char (*args, **kwargs)

Overloaded function.

1. set_char(self: rti.connexdds.DynamicData, name: str, value: str) -> None

Set a character value by name.

2. set_char(self: rti.connexdds.DynamicData, index: int, value: str) -> None

Set a character value by index.

3. set_char(self: rti.connexdds.DynamicData, name: str, value: int) -> None

Set a char value by field name using an int.

4. set_char(self: rti.connexdds.DynamicData, index: int, value: int) -> None

Set a char value by field index using an int.

set_char_values (*args, **kwargs)

Overloaded function.

1. set_char_values(self: rti.connexdds.DynamicData, name: str, values: object) -> None

Get multiple character values by field name.

2. set_char_values(self: rti.connexdds.DynamicData, index: int, values: object) -> None

Get multiple character values by field name.

set_complex (*args, **kwargs)

Overloaded function.

1. set_complex(self: rti.connexdds.DynamicData, name: str, value: rti.connexdds.DynamicData) -> None

Set a complex data type value by name.

2. set_complex(self: rti.connexdds.DynamicData, index: int, value: rti.connexdds.DynamicData) -> None

Set a complex data type value by index.

set_complex_values (*args, **kwargs)

Overloaded function.

1. set_complex_values(self: rti.connexdds.DynamicData, name: str, values: Iterable) -> None

Set a list of complex values by field name.

2. set_complex_values(self: rti.connexdds.DynamicData, index: int, values: Iterable) -> None

Set a list of complex values by index.

set_double (*args, **kwargs)

Overloaded function.

1. set_double(self: rti.connexdds.DynamicData, name: str, value: float) -> None

Set a 64-bit floating point value by name.

2. `set_double(self: rti.connextdds.DynamicData, index: int, value: float) -> None`

Set a 64-bit floating point value by index.

set_double_values (**args, **kwargs*)

Overloaded function.

1. `set_double_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 64-bit floating point values by field name.

2. `set_double_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 64-bit floating point values by field name.

set_float (**args, **kwargs*)

Overloaded function.

1. `set_float(self: rti.connextdds.DynamicData, name: str, value: float) -> None`

Set a 32-bit floating point value by name.

2. `set_float(self: rti.connextdds.DynamicData, index: int, value: float) -> None`

Set a 32-bit floating point value by index.

set_float128 (**args, **kwargs*)

Overloaded function.

1. `set_float128(self: rti.connextdds.DynamicData, name: str, value: rti.connextdds.LongDouble) -> None`

Set a 128-bit floating point value by name.

2. `set_float128(self: rti.connextdds.DynamicData, index: int, value: rti.connextdds.LongDouble) -> None`

Set a 128-bit floating point value by index.

set_float32 (**args, **kwargs*)

Overloaded function.

1. `set_float32(self: rti.connextdds.DynamicData, name: str, value: float) -> None`

Set a 32-bit floating point value by name.

2. `set_float32(self: rti.connextdds.DynamicData, index: int, value: float) -> None`

Set a 32-bit floating point value by index.

set_float32_values (**args, **kwargs*)

Overloaded function.

1. `set_float32_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 32-bit floating point values by field name.

2. `set_float32_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 32-bit floating point values by field name.

set_float64 (*args, **kwargs)

Overloaded function.

1. set_float64(self: rti.connextdds.DynamicData, name: str, value: float) -> None

Set a 64-bit floating point value by name.

2. set_float64(self: rti.connextdds.DynamicData, index: int, value: float) -> None

Set a 64-bit floating point value by index.

set_float64_values (*args, **kwargs)

Overloaded function.

1. set_float64_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None

Get multiple 64-bit floating point values by field name.

2. set_float64_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None

Get multiple 64-bit floating point values by field name.

set_float_values (*args, **kwargs)

Overloaded function.

1. set_float_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None

Get multiple 32-bit floating point values by field name.

2. set_float_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None

Get multiple 32-bit floating point values by field name.

set_int (*args, **kwargs)

Overloaded function.

1. set_int(self: rti.connextdds.DynamicData, name: str, value: int) -> None

Set a int (signed) value by name.

2. set_int(self: rti.connextdds.DynamicData, index: int, value: int) -> None

Set a int (signed) value by index.

set_int16 (*args, **kwargs)

Overloaded function.

1. set_int16(self: rti.connextdds.DynamicData, name: str, value: int) -> None

Set a 16-bit signed int value by name.

2. set_int16(self: rti.connextdds.DynamicData, index: int, value: int) -> None

Set a 16-bit signed int value by index.

set_int16_values (*args, **kwargs)

Overloaded function.

1. set_int16_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None

Get multiple 16-bit signed int values by field name.

2. `set_int16_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 16-bit signed int values by field name.

set_int32 (*args, **kwargs)

Overloaded function.

1. `set_int32(self: rti.connextdds.DynamicData, name: str, value: int) -> None`

Set a 32-bit signed int value by name.

2. `set_int32(self: rti.connextdds.DynamicData, index: int, value: int) -> None`

Set a 32-bit signed int value by index.

set_int32_values (*args, **kwargs)

Overloaded function.

1. `set_int32_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 32-bit signed int values by field name.

2. `set_int32_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 32-bit signed int values by field name.

set_int64 (*args, **kwargs)

Overloaded function.

1. `set_int64(self: rti.connextdds.DynamicData, name: str, value: int) -> None`

Set a 64-bit signed int value by name.

2. `set_int64(self: rti.connextdds.DynamicData, index: int, value: int) -> None`

Set a 64-bit signed int value by index.

set_int64_values (*args, **kwargs)

Overloaded function.

1. `set_int64_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 64-bit signed int values by field name.

2. `set_int64_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 64-bit signed int values by field name.

set_int8 (*args, **kwargs)

Overloaded function.

1. `set_int8(self: rti.connextdds.DynamicData, name: str, value: int) -> None`

Set an int8 value by field name using an int.

2. `set_int8(self: rti.connextdds.DynamicData, index: int, value: int) -> None`

Set an int8 value by field index using an int.

3. `set_int8(self: rti.connextdds.DynamicData, name: str, value: str) -> None`

Set an int8 value by field name using a char.

4. `set_int8(self: rti.connextdds.DynamicData, index: int, value: str) -> None`

Set an int8 value by field index using a char.

set_int_values (*args, **kwargs)

Overloaded function.

1. `set_int_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 32-bit signed int values by field name.

2. `set_int_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 32-bit signed int values by field name.

set_long (*args, **kwargs)

Overloaded function.

1. `set_long(self: rti.connextdds.DynamicData, name: str, value: int) -> None`

Set a 32-bit signed int value by name.

2. `set_long(self: rti.connextdds.DynamicData, index: int, value: int) -> None`

Set a 32-bit signed int value by index.

set_long_values (*args, **kwargs)

Overloaded function.

1. `set_long_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 32-bit signed int values by field name.

2. `set_long_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 32-bit signed int values by field name.

set_longdouble (*args, **kwargs)

Overloaded function.

1. `set_longdouble(self: rti.connextdds.DynamicData, name: str, value: rti.connextdds.Long-Double) -> None`

Set a 128-bit floating point value by name.

2. `set_longdouble(self: rti.connextdds.DynamicData, index: int, value: rti.connextdds.Long-Double) -> None`

Set a 128-bit floating point value by index.

set_longlong (*args, **kwargs)

Overloaded function.

1. `set_longlong(self: rti.connextdds.DynamicData, name: str, value: int) -> None`

Set a 64-bit signed int value by name.

2. `set_longlong(self: rti.connextdds.DynamicData, index: int, value: int) -> None`

Set a 64-bit signed int value by index.

set_longlong_values (**args, **kwargs*)

Overloaded function.

1. `set_longlong_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 64-bit signed int values by field name.

2. `set_longlong_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 64-bit signed int values by field name.

set_octet (**args, **kwargs*)

Overloaded function.

1. `set_octet(self: rti.connextdds.DynamicData, name: str, value: int) -> None`

Set a 8-bit unsigned int value by name.

2. `set_octet(self: rti.connextdds.DynamicData, index: int, value: int) -> None`

Set a 8-bit unsigned int value by index.

set_octet_values (**args, **kwargs*)

Overloaded function.

1. `set_octet_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 8-bit unsigned int values by field name.

2. `set_octet_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 8-bit unsigned int values by field name.

set_short (**args, **kwargs*)

Overloaded function.

1. `set_short(self: rti.connextdds.DynamicData, name: str, value: int) -> None`

Set a 16-bit signed int value by name.

2. `set_short(self: rti.connextdds.DynamicData, index: int, value: int) -> None`

Set a 16-bit signed int value by index.

set_short_values (**args, **kwargs*)

Overloaded function.

1. `set_short_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 16-bit signed int values by field name.

2. `set_short_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 16-bit signed int values by field name.

set_string (*args, **kwargs)

Overloaded function.

1. set_string(self: rti.connextdds.DynamicData, name: str, value: str) -> None

Set a string value by name.

2. set_string(self: rti.connextdds.DynamicData, index: int, value: str) -> None

Set a string value by index.

set_uint (*args, **kwargs)

Overloaded function.

1. set_uint(self: rti.connextdds.DynamicData, name: str, value: int) -> None

Set a 32-bit unsigned int value by name.

2. set_uint(self: rti.connextdds.DynamicData, index: int, value: int) -> None

Set a 32-bit unsigned int value by index.

set_uint16 (*args, **kwargs)

Overloaded function.

1. set_uint16(self: rti.connextdds.DynamicData, name: str, value: int) -> None

Set a 16-bit unsigned int value by name.

2. set_uint16(self: rti.connextdds.DynamicData, index: int, value: int) -> None

Set a 16-bit unsigned int value by index.

set_uint16_values (*args, **kwargs)

Overloaded function.

1. set_uint16_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None

Get multiple 16-bit unsigned int values by field name.

2. set_uint16_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None

Get multiple 16-bit unsigned int values by field name.

set_uint32 (*args, **kwargs)

Overloaded function.

1. set_uint32(self: rti.connextdds.DynamicData, name: str, value: int) -> None

Set a 32-bit unsigned int value by name.

2. set_uint32(self: rti.connextdds.DynamicData, index: int, value: int) -> None

Set a 32-bit unsigned int value by index.

set_uint32_values (*args, **kwargs)

Overloaded function.

1. set_uint32_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None

Get multiple 32-bit unsigned int values by field name.

2. `set_uint32_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 32-bit unsigned int values by field name.

set_uint64 (*args, **kwargs)

Overloaded function.

1. `set_uint64(self: rti.connextdds.DynamicData, name: str, value: int) -> None`

Set a 64-bit unsigned int value by name.

2. `set_uint64(self: rti.connextdds.DynamicData, index: int, value: int) -> None`

Set a 64-bit unsigned int value by index.

set_uint64_values (*args, **kwargs)

Overloaded function.

1. `set_uint64_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 64-bit unsigned int values by field name.

2. `set_uint64_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 64-bit unsigned int values by field name.

set_uint8 (*args, **kwargs)

Overloaded function.

1. `set_uint8(self: rti.connextdds.DynamicData, name: str, value: int) -> None`

Set a 8-bit unsigned int value by name.

2. `set_uint8(self: rti.connextdds.DynamicData, index: int, value: int) -> None`

Set a 8-bit unsigned int value by index.

set_uint8_values (*args, **kwargs)

Overloaded function.

1. `set_uint8_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 8-bit unsigned int values by field name.

2. `set_uint8_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 8-bit unsigned int values by field name.

set_uint_values (*args, **kwargs)

Overloaded function.

1. `set_uint_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None`

Get multiple 32-bit unsigned int values by field name.

2. `set_uint_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 32-bit unsigned int values by field name.

set_ulong (*args, **kwargs)

Overloaded function.

1. set_ulong(self: rti.connextdds.DynamicData, name: str, value: int) -> None

Set a 32-bit unsigned int value by name.

2. set_ulong(self: rti.connextdds.DynamicData, index: int, value: int) -> None

Set a 32-bit unsigned int value by index.

set_ulong_values (*args, **kwargs)

Overloaded function.

1. set_ulong_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None

Get multiple 32-bit unsigned int values by field name.

2. set_ulong_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None

Get multiple 32-bit unsigned int values by field name.

set_ulonglong (*args, **kwargs)

Overloaded function.

1. set_ulonglong(self: rti.connextdds.DynamicData, name: str, value: int) -> None

Set a 64-bit unsigned int value by name.

2. set_ulonglong(self: rti.connextdds.DynamicData, index: int, value: int) -> None

Set a 64-bit unsigned int value by index.

set_ulonglong_values (*args, **kwargs)

Overloaded function.

1. set_ulonglong_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None

Get multiple 64-bit unsigned int values by field name.

2. set_ulonglong_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None

Get multiple 64-bit unsigned int values by field name.

set_ushort (*args, **kwargs)

Overloaded function.

1. set_ushort(self: rti.connextdds.DynamicData, name: str, value: int) -> None

Set a 16-bit unsigned int value by name.

2. set_ushort(self: rti.connextdds.DynamicData, index: int, value: int) -> None

Set a 16-bit unsigned int value by index.

set_ushort_values (*args, **kwargs)

Overloaded function.

1. set_ushort_values(self: rti.connextdds.DynamicData, name: str, values: object) -> None

Get multiple 16-bit unsigned int values by field name.

2. `set_ushort_values(self: rti.connextdds.DynamicData, index: int, values: object) -> None`

Get multiple 16-bit unsigned int values by field name.

set_value (*args, **kwargs)

Overloaded function.

1. `set_value(self: rti.connextdds.DynamicData, field_path: str, value: object) -> None`

Automatically resolve type and set value for a field.

2. `set_value(self: rti.connextdds.DynamicData, field_path: int, value: object) -> None`

Automatically resolve type and set value for a field.

set_values (*args, **kwargs)

Overloaded function.

1. `set_values(self: rti.connextdds.DynamicData, field_path: str, values: object) -> None`

Automatically resolve type and set collection for a field.

2. `set_values(self: rti.connextdds.DynamicData, field_path: int, values: object) -> None`

Automatically resolve type and set collection for a field.

set_wchar (*args, **kwargs)

Overloaded function.

1. `set_wchar(self: rti.connextdds.DynamicData, name: str, value: str) -> None`

Set a wchar value by field name.

2. `set_wchar(self: rti.connextdds.DynamicData, index: int, value: str) -> None`

Set a wchar value by field index.

set_wstring (*args, **kwargs)

Overloaded function.

1. `set_wstring(self: rti.connextdds.DynamicData, name: str, value: str) -> None`

Set a wstring by field name.

2. `set_wstring(self: rti.connextdds.DynamicData, index: int, value: str) -> None`

Set a wstring value by field index.

to_cdr_buffer (self: rti.connextdds.DynamicData) → *rti.connextdds.CharSeq*

Serializes a DynamicData sample to CDR format

to_json (self: rti.connextdds.DynamicData) → str

Convert DynamicData object to a JSON string representation

to_string (*self*: rti.connexdds.DynamicData, *format*: rti.connexdds.PrintFormatProperty = *PrintFormatProperty.default*) → str

Convert DynamicData object to string with print format.

property type

Gets the data type of this DynamicData.

property type_kind

Gets this data type kind of this DynamicData.

update (*self*: rti.connexdds.DynamicData, *arg0*: dict) → None

class rti.connexdds.DynamicDataEncapsulationKind

Bases: pybind11_object

CDR_BIG_ENDIAN = <DynamicDataEncapsulationKind.CDR_BIG_ENDIAN: 0>

CDR_LITTLE_ENDIAN =

<DynamicDataEncapsulationKind.CDR_LITTLE_ENDIAN: 1>

DEFAULT = <DynamicDataEncapsulationKind.DEFAULT: 2147483647>

class DynamicDataEncapsulationKind

Bases: pybind11_object

Members:

CDR_BIG_ENDIAN : CDR big endian encapsulation.

CDR_LITTLE_ENDIAN : CDR little endian encapsulation.

PL_CDR_BIG_ENDIAN : PL CDR big endian encapsulation.

PL_CDR_LITTLE_ENDIAN : PL CDR little endian encapsulation.

DEFAULT : Default encapsulation.

CDR_BIG_ENDIAN = <DynamicDataEncapsulationKind.CDR_BIG_ENDIAN: 0>

CDR_LITTLE_ENDIAN =

<DynamicDataEncapsulationKind.CDR_LITTLE_ENDIAN: 1>

DEFAULT = <DynamicDataEncapsulationKind.DEFAULT: 2147483647>

PL_CDR_BIG_ENDIAN =

<DynamicDataEncapsulationKind.PL_CDR_BIG_ENDIAN: 2>

PL_CDR_LITTLE_ENDIAN =

<DynamicDataEncapsulationKind.PL_CDR_LITTLE_ENDIAN: 3>

__eq__ (*self*: object, *other*: object) → bool

__getstate__ (*self*: object) → int


```

__hash__(self: object) → int

__index__(self: rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind) →
    int

__init__(self:
    rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind,
    value: int) → None

__int__(self:
    rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind)
    → int

__members__ = {'CDR_BIG_ENDIAN':
<DynamicDataEncapsulationKind.CDR_BIG_ENDIAN: 0>,
'CDR_LITTLE_ENDIAN':
<DynamicDataEncapsulationKind.CDR_LITTLE_ENDIAN: 1>, 'DEFAULT':
<DynamicDataEncapsulationKind.DEFAULT: 2147483647>,
'PL_CDR_BIG_ENDIAN':
<DynamicDataEncapsulationKind.PL_CDR_BIG_ENDIAN: 2>,
'PL_CDR_LITTLE_ENDIAN':
<DynamicDataEncapsulationKind.PL_CDR_LITTLE_ENDIAN: 3>}

__module__ = 'rti.connextdds'

__ne__(self: object, other: object) → bool

__repr__(self: object) → str

__setstate__(self: rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind, state: int) →
    None

__str__()
    name(self: handle) -> str

property name

property value

PL_CDR_BIG_ENDIAN =
<DynamicDataEncapsulationKind.PL_CDR_BIG_ENDIAN: 2>

PL_CDR_LITTLE_ENDIAN =
<DynamicDataEncapsulationKind.PL_CDR_LITTLE_ENDIAN: 3>

__eq__(self: rti.connextdds.DynamicDataEncapsulationKind, arg0:
    rti.connextdds.DynamicDataEncapsulationKind) → bool

Apply operator to underlying enumerated values.

```

__ge__ (*self*: rti.connexdds.DynamicDataEncapsulationKind, *arg0*: rti.connexdds.DynamicDataEncapsulationKind) → bool

Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.DynamicDataEncapsulationKind, *arg0*: rti.connexdds.DynamicDataEncapsulationKind) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DynamicDataEncapsulationKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.DynamicDataEncapsulationKind, *arg0*: rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.DynamicDataEncapsulationKind) → *rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind*

Int conversion.

__le__ (*self*: rti.connexdds.DynamicDataEncapsulationKind, *arg0*: rti.connexdds.DynamicDataEncapsulationKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.DynamicDataEncapsulationKind, *arg0*: rti.connexdds.DynamicDataEncapsulationKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.DynamicDataEncapsulationKind, *arg0*: rti.connexdds.DynamicDataEncapsulationKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.DynamicDataEncapsulationKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.**DynamicDataInfo**

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.DynamicDataInfo, *arg0*: rti.connexdds.DynamicDataInfo) → bool

Test for equality.

```

__hash__ = None

__init__ (*args, **kwargs)

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.DynamicDataInfo, arg0: rti.connexdds.DynamicDataInfo) → bool
    Test for inequality.

property encapsulation_kind
    The encapsulation kind.

property is_optimized_storage
    Flag indicating whether storage is optimized.

property member_count
    The number of members of this sample.

property stored_size
    The number of bytes currently used to store the data of this sample.

class rti.connexdds.DynamicDataMemberInfo
    Bases: pybind11_object

    __eq__ (self: rti.connexdds.DynamicDataMemberInfo, arg0:
        rti.connexdds.DynamicDataMemberInfo) → bool
        Test for equality.

    __hash__ = None

    __init__ (self: rti.connexdds.DynamicDataMemberInfo) → None
        Create a DynamicDataMemberInfo object.

    __module__ = 'rti.connexdds'

    __ne__ (self: rti.connexdds.DynamicDataMemberInfo, arg0:
        rti.connexdds.DynamicDataMemberInfo) → bool
        Test for inequality.

property element_count
    The number of elements in the member.

property element_kind
    The type kind of the elements in the member

property index
    The member index

property kind
    The member type kind

```

property name

The member name

class `rti.connexdds.DynamicDataProperty`Bases: `pybind11_object`**__eq__** (*self*: `rti.connexdds.DynamicDataProperty`, *arg0*: `rti.connexdds.DynamicDataProperty`) → `bool`

Test for equality.

__hash__ = `None`**__init__** (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: `rti.connexdds.DynamicDataProperty`) -> `None`

Construct with default settings.

2. **__init__**(*self*: `rti.connexdds.DynamicDataProperty`, *initial_buffer_size*: `int`, *max_buffer_size*: `int`, *max_buffer_size_increment*: `int`, *check_buffer_size*: `bool`) -> `None`**__module__** = `'rti.connexdds'`**__ne__** (*self*: `rti.connexdds.DynamicDataProperty`, *arg0*: `rti.connexdds.DynamicDataProperty`) → `bool`

Test for inequality.

property `check_buffer_size`

Check buffer size.

property `initial_buffer_size`

Initial buffer size.

property `max_buffer_size`

Max buffer size.

property `max_buffer_size_increment`

Max buffer size.

class `rti.connexdds.DynamicDataSeq`Bases: `pybind11_object`**__add__** (*self*: `rti.connexdds.DynamicDataSeq`, *arg0*: `rti.connexdds.DynamicDataSeq`) → `rti.connexdds.DynamicDataSeq`**__bool__** (*self*: `rti.connexdds.DynamicDataSeq`) → `bool`

Check whether the list is nonempty

__contains__ (*self*: `rti.connexdds.DynamicDataSeq`, *x*: `rti.connexdds.DynamicData`) → `bool`Return true the container contains *x*

`__delitem__` (*args, **kwargs)

Overloaded function.

1. `__delitem__(self: rti.connextdds.DynamicDataSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connextdds.DynamicDataSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (self: rti.connextdds.DynamicDataSeq, arg0: rti.connextdds.DynamicDataSeq) → bool

`__getitem__` (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connextdds.DynamicDataSeq, s: slice) -> rti.connextdds.DynamicDataSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.DynamicDataSeq, arg0: int) -> rti.connextdds.DynamicData`

`__hash__` = None

`__iadd__` (self: rti.connextdds.DynamicDataSeq, arg0: rti.connextdds.DynamicDataSeq) → *rti.connextdds.DynamicDataSeq*

`__imul__` (self: rti.connextdds.DynamicDataSeq, arg0: int) → *rti.connextdds.DynamicDataSeq*

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.DynamicDataSeq) -> None`
2. `__init__(self: rti.connextdds.DynamicDataSeq, arg0: rti.connextdds.DynamicDataSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.DynamicDataSeq, arg0: Iterable) -> None`

`__iter__` (self: rti.connextdds.DynamicDataSeq) → Iterator

`__len__` (self: rti.connextdds.DynamicDataSeq) → int

`__module__` = 'rti.connextdds'

`__mul__` (self: rti.connextdds.DynamicDataSeq, arg0: int) → *rti.connextdds.DynamicDataSeq*

`__ne__` (self: rti.connextdds.DynamicDataSeq, arg0: rti.connextdds.DynamicDataSeq) → bool

`__repr__` (self: rti.connextdds.DynamicDataSeq) → str

Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.DynamicDataSeq, *arg0*: int) → *rti.connexdds.DynamicDataSeq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.DynamicDataSeq, *arg0*: int, *arg1*: rti.connexdds.DynamicData) -> None
2. **__setitem__**(*self*: rti.connexdds.DynamicDataSeq, *arg0*: slice, *arg1*: rti.connexdds.DynamicDataSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.DynamicDataSeq, *x*: rti.connexdds.DynamicData) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.DynamicDataSeq) → None

Clear the contents

count (*self*: rti.connexdds.DynamicDataSeq, *x*: rti.connexdds.DynamicData) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.DynamicDataSeq, *L*: rti.connexdds.DynamicDataSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.DynamicDataSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.DynamicDataSeq, *i*: int, *x*: rti.connexdds.DynamicData) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.DynamicDataSeq) -> rti.connexdds.DynamicData

Remove and return the last item

2. **pop**(*self*: rti.connexdds.DynamicDataSeq, *i*: int) -> rti.connexdds.DynamicData

Remove and return the item at index *i*

remove (*self*: rti.connexdds.DynamicDataSeq, *x*: rti.connexdds.DynamicData) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.DynamicDataTimestampedSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: rti.connexdds.DynamicDataTimestampedSeq) → *rti.connexdds.DynamicDataTimestampedSeq*

__bool__ (*self*: rti.connexdds.DynamicDataTimestampedSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.DynamicDataTimestampedSeq, *x*:
Tuple[rti.connexdds.DynamicData, rti.connexdds.Time]) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*:
rti.connexdds.DynamicDataTimestampedSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.DynamicDataTimestampedSeq, *s*: slice) -> rti.connexdds.DynamicDataTimestampedSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: int) -> *Tuple*[rti.connexdds.DynamicData, rti.connexdds.Time]

__hash__ = None

__iadd__ (*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*:
rti.connexdds.DynamicDataTimestampedSeq) →
rti.connexdds.DynamicDataTimestampedSeq

__imul__ (*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: int) →
rti.connexdds.DynamicDataTimestampedSeq

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.DynamicDataTimestampedSeq) -> None

2. **__init__**(*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: rti.connexdds.DynamicDataTimestampedSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.DynamicDataTimestampedSeq) → Iterator

__len__ (*self*: rti.connexdds.DynamicDataTimestampedSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: int) →
rti.connexdds.DynamicDataTimestampedSeq

__ne__ (*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*:
 rti.connexdds.DynamicDataTimestampedSeq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule
 object NULL>

__rmul__ (*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: int) →
rti.connexdds.DynamicDataTimestampedSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: int, *arg1*: Tuple[rti.connexdds.DynamicData, rti.connexdds.Time]) -> None
2. **__setitem__**(*self*: rti.connexdds.DynamicDataTimestampedSeq, *arg0*: slice, *arg1*: rti.connexdds.DynamicDataTimestampedSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.DynamicDataTimestampedSeq, *x*: Tuple[rti.connexdds.DynamicData, rti.connexdds.Time]) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.DynamicDataTimestampedSeq) → None

Clear the contents

count (*self*: rti.connexdds.DynamicDataTimestampedSeq, *x*: Tuple[rti.connexdds.DynamicData, rti.connexdds.Time]) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.DynamicDataTimestampedSeq, *L*: rti.connexdds.DynamicDataTimestampedSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.DynamicDataTimestampedSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.DynamicDataTimestampedSeq, *i*: int, *x*:
 Tuple[rti.connexdds.DynamicData, rti.connexdds.Time]) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connexdds.DynamicDataTimestampedSeq) -> Tuple[rti.connexdds.DynamicData, rti.connexdds.Time]

Remove and return the last item

2. pop(self: rti.connexdds.DynamicDataTimestampedSeq, i: int) -> Tuple[rti.connexdds.DynamicData, rti.connexdds.Time]

Remove and return the item at index i

remove (self: rti.connexdds.DynamicDataTimestampedSeq, x: Tuple[rti.connexdds.DynamicData, rti.connexdds.Time]) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class rti.connexdds.DynamicDataTypeSerializationProperty

Bases: pybind11_object

DEFAULT = <rti.connexdds.DynamicDataTypeSerializationProperty object>

__eq__ (self: rti.connexdds.DynamicDataTypeSerializationProperty, arg0: rti.connexdds.DynamicDataTypeSerializationProperty) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. __init__(self: rti.connexdds.DynamicDataTypeSerializationProperty) -> None

Construct with default settings.

2. __init__(self: rti.connexdds.DynamicDataTypeSerializationProperty, max_serialized_size: int, min_serialized_size: int, trim_to_size: bool, skip_deserialization: bool) -> None

Specify all serialization settings at object creation time.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.DynamicDataTypeSerializationProperty, arg0: rti.connexdds.DynamicDataTypeSerializationProperty) → bool

Test for inequality.

property max_serialized_size

The maximum number of bytes that objects of a given type could consume when serialized on the network. [DEPRECATED]

property min_serialized_size

The minimum number of bytes that objects of a given type could consume when serialized on the network. [DEPRECATED]

property skip_deserialization

Controls whether the DynamicData object will be deserialized by a DynamicDataReader.

property trim_to_size

Controls the growth of the buffer in a DynamicData object.

class `rti.connexdds.DynamicType`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.DynamicType`, *arg0*: `rti.connexdds.DynamicType`) → bool

Compare DynamicType objects for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.DynamicType`, *arg0*: `rti.connexdds.DynamicType`) → bool

Compare DynamicType objects for inequality.

__str__ (*self*: `rti.connexdds.DynamicType`) → str

DynamicData value to string.

is_aggregation_type (*self*: `rti.connexdds.DynamicType`) → bool

Determines if this DynamicType is an aggregation type.

is_collection_type (*self*: `rti.connexdds.DynamicType`) → bool

Determines if this DynamicType is a CollectionType.

is_constructed_type (*self*: `rti.connexdds.DynamicType`) → bool

Determines if this DynamicType is a constructed type.

is_keyed (*self*: `rti.connexdds.DynamicType`) → bool

Determines if this DynamicType has a key.

is_primitive_type (*self*: `rti.connexdds.DynamicType`) → bool

Determines if this DynamicType is a PrimitiveType

property kind

Get the type kind.

property name

Gets the name.

print_idl (*self*: `rti.connexdds.DynamicType`, *index*: `int = 0`) → None

Prints the IDL representation of this type to the standard output.

to_string (*self*: `rti.connexdds.DynamicType`, *format*:
`rti.connexdds.DynamicTypePrintFormatProperty =`
`DynamicTypePrintFormatProperty()`) → str

Convert DynamicType to string with print format.

class `rti.connexdds.DynamicTypePrintFormatProperty`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.DynamicTypePrintFormatProperty`, *arg0*: `rti.connexdds.DynamicTypePrintFormatProperty`) → bool

Test for equality.

__hash__ = None

__init__ (*self*: `rti.connexdds.DynamicTypePrintFormatProperty`, *indent*: `int = 0`, *print_ordinals*: `bool = False`) → None

Construct with default settings.

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.DynamicTypePrintFormatProperty`, *arg0*: `rti.connexdds.DynamicTypePrintFormatProperty`) → bool

Test for inequality.

property indent

The amount of additional indent to be included when converting a `DynamicType` to a string.

property min_serialized_size

The minimum number of bytes that objects of a given type could consume when serialized on the network. [DEPRECATED]

class `rti.connexdds.EndpointGroup`

Bases: `pybind11_object`

__init__ (*self*: `rti.connexdds.EndpointGroup`, *role_name*: `str`, *quorum_count*: `int`) → None

Create an `EndpointGroup` with the provided parameters.

__module__ = `'rti.connexdds'`

property quorum_count

Get/set the `EndpointGroup`'s quorum count.

property role_name

Get/set the `EndpointGroup`'s role name.

class `rti.connexdds.EndpointGroupSeq`

Bases: `pybind11_object`

__add__ (*self*: `rti.connexdds.EndpointGroupSeq`, *arg0*: `rti.connexdds.EndpointGroupSeq`) → `rti.connexdds.EndpointGroupSeq`

__bool__ (*self*: `rti.connexdds.EndpointGroupSeq`) → bool

Check whether the list is nonempty

__contains__ (*self*: `rti.connexdds.EndpointGroupSeq`, *x*: `rti.connexdds.EndpointGroup`) → bool

Return true the container contains *x*

`__delitem__` (*args, **kwargs)

Overloaded function.

1. `__delitem__(self: rti.connexdds.EndpointGroupSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connexdds.EndpointGroupSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (self: rti.connexdds.EndpointGroupSeq, arg0: rti.connexdds.EndpointGroupSeq) → bool

`__getitem__` (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connexdds.EndpointGroupSeq, s: slice) -> rti.connexdds.EndpointGroupSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.EndpointGroupSeq, arg0: int) -> rti.connexdds.EndpointGroup`

`__hash__` = None

`__iadd__` (self: rti.connexdds.EndpointGroupSeq, arg0: rti.connexdds.EndpointGroupSeq) → *rti.connexdds.EndpointGroupSeq*

`__imul__` (self: rti.connexdds.EndpointGroupSeq, arg0: int) → *rti.connexdds.EndpointGroupSeq*

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.EndpointGroupSeq) -> None`
2. `__init__(self: rti.connexdds.EndpointGroupSeq, arg0: rti.connexdds.EndpointGroupSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.EndpointGroupSeq, arg0: Iterable) -> None`

`__iter__` (self: rti.connexdds.EndpointGroupSeq) → Iterator

`__len__` (self: rti.connexdds.EndpointGroupSeq) → int

`__module__` = 'rti.connexdds'

`__mul__` (self: rti.connexdds.EndpointGroupSeq, arg0: int) → *rti.connexdds.EndpointGroupSeq*

`__ne__` (self: rti.connexdds.EndpointGroupSeq, arg0: rti.connexdds.EndpointGroupSeq) → bool

`__rmul__` (self: rti.connexdds.EndpointGroupSeq, arg0: int) → *rti.connexdds.EndpointGroupSeq*

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connexdds.EndpointGroupSeq, arg0: int, arg1: rti.connexdds.EndpointGroup) -> None`
2. `__setitem__(self: rti.connexdds.EndpointGroupSeq, arg0: slice, arg1: rti.connexdds.EndpointGroupSeq) -> None`

Assign list elements using a slice object

append (self: rti.connexdds.EndpointGroupSeq, x: rti.connexdds.EndpointGroup) → None

Add an item to the end of the list

clear (self: rti.connexdds.EndpointGroupSeq) → None

Clear the contents

count (self: rti.connexdds.EndpointGroupSeq, x: rti.connexdds.EndpointGroup) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connexdds.EndpointGroupSeq, L: rti.connexdds.EndpointGroupSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.EndpointGroupSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connexdds.EndpointGroupSeq, i: int, x: rti.connexdds.EndpointGroup) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connexdds.EndpointGroupSeq) -> rti.connexdds.EndpointGroup`

Remove and return the last item

2. `pop(self: rti.connexdds.EndpointGroupSeq, i: int) -> rti.connexdds.EndpointGroup`

Remove and return the item at index i

remove (self: rti.connexdds.EndpointGroupSeq, x: rti.connexdds.EndpointGroup) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class rti.connexdds.**EndpointGroupVector**

Bases: pybind11_object

A DDS standard container with functionality similar to a C++ vector.

__eq__ (*self*: rti.connexdds.EndpointGroupVector, *arg0*: rti.connexdds.EndpointGroupVector) → bool

Compare EndpointGroupVectors for equality.

__getitem__ (*self*: rti.connexdds.EndpointGroupVector, *arg0*: int) → rti.connexdds.EndpointGroup

Get the value at the specified index.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.EndpointGroupVector) -> None

Create an empty EndpointGroupVector

2. **__init__**(*self*: rti.connexdds.EndpointGroupVector, *size*: int) -> None

Create a EndpointGroupVector with a preallocated size.

3. **__init__**(*self*: rti.connexdds.EndpointGroupVector, *vector*: rti.connexdds.EndpointGroupVector) -> None

Create a copy from another EndpointGroupVector.

4. **__init__**(*self*: rti.connexdds.EndpointGroupVector, *arg0*: Iterable) -> None

5. **__init__**(*self*: rti.connexdds.EndpointGroupVector, *list*: rti.connexdds.EndpointGroupSeq) -> None

Create a EndpointGroupVector from a list of values.

__iter__ (*self*: rti.connexdds.EndpointGroupVector) → Iterator

Iterate over the contents of the vector.

__len__ (*self*: rti.connexdds.EndpointGroupVector) → int

Get the length of the EndpointGroupVector.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.EndpointGroupVector, *arg0*: rti.connexdds.EndpointGroupVector) → bool

Compare EndpointGroupVectors for inequality.

__setitem__ (*self*: rti.connexdds.EndpointGroupVector, *arg0*: int, *arg1*: rti.connexdds.EndpointGroup) → None

Set the value at the specified index.

clear (*self*: rti.connexdds.EndpointGroupVector) → None

Resize EndpointGroupVector to 0.

resize (*self*: rti.connexdds.EndpointGroupVector, *size*: int) → None

Resize EndpointGroupVector.

```

class rti.connextdds.Entity
    Bases: IEntity
    __init__ (*args, **kwargs)
    __module__ = 'rti.connextdds'

class rti.connextdds.EntityFactory
    Bases: pybind11_object
    __eq__ (self: rti.connextdds.EntityFactory, arg0: rti.connextdds.EntityFactory) → bool
        Test for equality.
    __hash__ = None
    __init__ (*args, **kwargs)
        Overloaded function.
        1. __init__(self: rti.connextdds.EntityFactory) -> None
            Creates the default policy.
        2. __init__(self: rti.connextdds.EntityFactory, auto_enable: bool) -> None
            Specifies whether the entity acting as a factory automatically enables the instances it creates.
    __module__ = 'rti.connextdds'
    __ne__ (self: rti.connextdds.EntityFactory, arg0: rti.connextdds.EntityFactory) → bool
        Test for inequality.
    auto_enable = <rti.connextdds.EntityFactory object>
    property autoenable_created_entities
        Whether the entity acting as a factory automatically enables the instances it creates.
    manually_enable = <rti.connextdds.EntityFactory object>

class rti.connextdds.EntityName
    Bases: pybind11_object
    __eq__ (self: rti.connextdds.EntityName, arg0: rti.connextdds.EntityName) → bool
        Test for equality.
    __hash__ = None
    __init__ (*args, **kwargs)
        Overloaded function.
        1. __init__(self: rti.connextdds.EntityName) -> None
            Creates the default policy (no name).
        2. __init__(self: rti.connextdds.EntityName, name: str) -> None
            Creates an instance that specifies an entity name.

```

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.EntityName, *arg0*: rti.connexdds.EntityName) → bool
Test for inequality.

property name

Sets the entity name.

property role_name

Sets the role name.

class rti.connexdds.**EntitySeq**

Bases: pybind11_object

__add__ (*self*: rti.connexdds.EntitySeq, *arg0*: rti.connexdds.EntitySeq) → *rti.connexdds.EntitySeq*

__bool__ (*self*: rti.connexdds.EntitySeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.EntitySeq, *x*: rti.connexdds.Entity) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.EntitySeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.EntitySeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.EntitySeq, *arg0*: rti.connexdds.EntitySeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.EntitySeq, *s*: slice) -> rti.connexdds.EntitySeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.EntitySeq, *arg0*: int) -> rti.connexdds.Entity

__hash__ = None

__iadd__ (*self*: rti.connexdds.EntitySeq, *arg0*: rti.connexdds.EntitySeq) →
rti.connexdds.EntitySeq

__imul__ (*self*: rti.connexdds.EntitySeq, *arg0*: int) → *rti.connexdds.EntitySeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.EntitySeq) -> None

2. `__init__(self: rti.connextdds.EntitySeq, arg0: rti.connextdds.EntitySeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.EntitySeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.EntitySeq) -> Iterator`

`__len__(self: rti.connextdds.EntitySeq) -> int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.EntitySeq, arg0: int) -> rti.connextdds.EntitySeq`

`__ne__(self: rti.connextdds.EntitySeq, arg0: rti.connextdds.EntitySeq) -> bool`

`__rmul__(self: rti.connextdds.EntitySeq, arg0: int) -> rti.connextdds.EntitySeq`

`__setitem__(*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connextdds.EntitySeq, arg0: int, arg1: rti.connextdds.Entity) -> None`

2. `__setitem__(self: rti.connextdds.EntitySeq, arg0: slice, arg1: rti.connextdds.EntitySeq) -> None`

Assign list elements using a slice object

`append(self: rti.connextdds.EntitySeq, x: rti.connextdds.Entity) -> None`

Add an item to the end of the list

`clear(self: rti.connextdds.EntitySeq) -> None`

Clear the contents

`count(self: rti.connextdds.EntitySeq, x: rti.connextdds.Entity) -> int`

Return the number of times `x` appears in the list

`extend(*args, **kwargs)`

Overloaded function.

1. `extend(self: rti.connextdds.EntitySeq, L: rti.connextdds.EntitySeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.EntitySeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

`insert(self: rti.connextdds.EntitySeq, i: int, x: rti.connextdds.Entity) -> None`

Insert an item at a given position.

`pop(*args, **kwargs)`

Overloaded function.

1. `pop(self: rti.connextdds.EntitySeq) -> rti.connextdds.Entity`

Remove and return the last item

2. `pop(self: rti.connextdds.EntitySeq, i: int) -> rti.connextdds.Entity`

Remove and return the item at index `i`

`remove(self: rti.connextdds.EntitySeq, x: rti.connextdds.Entity) → None`

Remove the first item from the list whose value is `x`. It is an error if there is no such item.

class `rti.connextdds.EnumMember`

Bases: `pybind11_object`

`__eq__(*args, **kwargs)`

Overloaded function.

1. `__eq__(self: rti.connextdds.EnumMember, arg0: rti.connextdds.EnumMember) -> bool`

Test for equality.

2. `__eq__(self: rti.connextdds.EnumMember, arg0: rti.connextdds.EnumMember) -> bool`

Test for equality.

3. `__eq__(self: rti.connextdds.EnumMember, arg0: int) -> bool`

Test for equality.

`__hash__ = None`

`__init__(self: rti.connextdds.EnumMember, name: str, ordinal: int) → None`

Create an `EnumMember` with a given name and ordinal value.

`__int__(self: rti.connextdds.EnumMember) → int`

`__long__(self: rti.connextdds.EnumMember) → int`

`__module__ = 'rti.connextdds'`

`__ne__(*args, **kwargs)`

Overloaded function.

1. `__ne__(self: rti.connextdds.EnumMember, arg0: rti.connextdds.EnumMember) -> bool`

Test for inequality.

2. `__ne__(self: rti.connextdds.EnumMember, arg0: int) -> bool`

Test for inequality.

`__str__(self: rti.connextdds.EnumMember) → str`

property `name`

The member name.

property `ordinal`

The member's ordinal.

class `rti.connextdds.EnumMemberSeq`

Bases: `pybind11_object`

__add__ (*self*: rti.connexdds.EnumMemberSeq, *arg0*: rti.connexdds.EnumMemberSeq) → *rti.connexdds.EnumMemberSeq*

__bool__ (*self*: rti.connexdds.EnumMemberSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.EnumMemberSeq, *x*: rti.connexdds.EnumMember) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.EnumMemberSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.EnumMemberSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.EnumMemberSeq, *arg0*: rti.connexdds.EnumMemberSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.EnumMemberSeq, *s*: slice) -> rti.connexdds.EnumMemberSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.EnumMemberSeq, *arg0*: int) -> rti.connexdds.EnumMember

__hash__ = None

__iadd__ (*self*: rti.connexdds.EnumMemberSeq, *arg0*: rti.connexdds.EnumMemberSeq) → *rti.connexdds.EnumMemberSeq*

__imul__ (*self*: rti.connexdds.EnumMemberSeq, *arg0*: int) → *rti.connexdds.EnumMemberSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.EnumMemberSeq) -> None

2. **__init__**(*self*: rti.connexdds.EnumMemberSeq, *arg0*: rti.connexdds.EnumMemberSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.EnumMemberSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.EnumMemberSeq) → Iterator

__len__ (*self*: rti.connexdds.EnumMemberSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.EnumMemberSeq, *arg0*: int) → rti.connexdds.EnumMemberSeq

__ne__ (*self*: rti.connexdds.EnumMemberSeq, *arg0*: rti.connexdds.EnumMemberSeq) → bool

__rmul__ (*self*: rti.connexdds.EnumMemberSeq, *arg0*: int) → rti.connexdds.EnumMemberSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.EnumMemberSeq, *arg0*: int, *arg1*: rti.connexdds.EnumMember) -> None
2. **__setitem__**(*self*: rti.connexdds.EnumMemberSeq, *arg0*: slice, *arg1*: rti.connexdds.EnumMemberSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.EnumMemberSeq, *x*: rti.connexdds.EnumMember) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.EnumMemberSeq) → None

Clear the contents

count (*self*: rti.connexdds.EnumMemberSeq, *x*: rti.connexdds.EnumMember) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.EnumMemberSeq, *L*: rti.connexdds.EnumMemberSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.EnumMemberSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.EnumMemberSeq, *i*: int, *x*: rti.connexdds.EnumMember) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.EnumMemberSeq) -> rti.connexdds.EnumMember

Remove and return the last item

2. **pop**(*self*: rti.connexdds.EnumMemberSeq, *i*: int) -> rti.connexdds.EnumMember

Remove and return the item at index *i*

remove (*self*: rti.connexdds.EnumMemberSeq, *x*: rti.connexdds.EnumMember) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class `rti.connexdds.EnumType`

Bases: *ACTEnumMember*

`__eq__` (*self*: `rti.connexdds.EnumType`, *arg0*: `rti.connexdds.EnumType`) → bool

Test for equality.

`__hash__` = None

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__`(*self*: `rti.connexdds.EnumType`, *name*: str) -> None

Creates an empty EnumType.

2. `__init__`(*self*: `rti.connexdds.EnumType`, *name*: str, *members*: `rti.connexdds.EnumMemberSeq`) -> None

Creates a enum with the members in the list.

3. `__init__`(*self*: `rti.connexdds.EnumType`, *type*: `rti.connexdds.DynamicType`) -> None

Cast a DynamicType to an EnumType.

`__module__` = `'rti.connexdds'`

`__ne__` (*self*: `rti.connexdds.EnumType`, *arg0*: `rti.connexdds.EnumType`) → bool

Test for inequality.

`add_member` (*self*: `rti.connexdds.EnumType`, *member*: `rti.connexdds.EnumMember`) → `rti.connexdds.EnumType`

Adds a member at the end.

`add_members` (*self*: `rti.connexdds.EnumType`, *members*: `rti.connexdds.EnumMemberSeq`) → `rti.connexdds.EnumType`

Adds a member at the end.

`property extensibility_kind`

Enum's extensibility kind.

`find_member_by_ordinal` (*self*: `rti.connexdds.EnumType`, *ordinal*: int) → int

Gets the index of the member with this ordinal value.

exception `rti.connexdds.Error`

Bases: *Exception*

`__module__` = `'rti.connexdds'`

class `rti.connexdds.Event`

Bases: *pybind11_object*

`__eq__` (*self*: `rti.connexdds.Event`, *arg0*: `rti.connexdds.Event`) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connextdds.Event) -> None

Creates the default policy.

2. **__init__**(self: rti.connextdds.Event, thread: rti.connextdds.ThreadSettings, initial_count: int, max_count: int) -> None

Creates an instance with all the parameters set.

__module__ = 'rti.connextdds'

__ne__(self: rti.connextdds.Event, arg0: rti.connextdds.Event) → bool

Test for inequality.

property initial_count

Event thread QoS.

property max_count

Event thread QoS.

property thread

Event thread QoS.

class rti.connextdds.**EventCount32**

Bases: pybind11_object

__init__ (*args, **kwargs)

__module__ = 'rti.connextdds'

property change

The incremental count.

property total

The total count.

class rti.connextdds.**EventCount64**

Bases: pybind11_object

__init__ (*args, **kwargs)

__module__ = 'rti.connextdds'

property change

The incremental count.

property total

The total count.

exception `rti.connexdds.Exception`

Bases: *Exception*

__module__ = `'rti.connexdds'`

__weakref__

list of weak references to the object (if defined)

class `rti.connexdds.ExclusiveArea`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.ExclusiveArea`, *arg0*: `rti.connexdds.ExclusiveArea`) → `bool`

Test for equality.

__hash__ = `None`

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.ExclusiveArea) -> None`

Creates the default policy.

2. `__init__(self: rti.connexdds.ExclusiveArea, use_shared_exclusive_area: bool) -> None`

Creates an instance specifying the use of shared or exclusive area.

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.ExclusiveArea`, *arg0*: `rti.connexdds.ExclusiveArea`) → `bool`

Test for inequality.

exclusive_ea = `<rti.connexdds.ExclusiveArea object>`

shared_ea = `<rti.connexdds.ExclusiveArea object>`

property use_shared_exclusive_area

Whether the Entity is protected by its own exclusive area or the shared one.

class `rti.connexdds.ExpressionProperty`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.ExpressionProperty`, *arg0*: `rti.connexdds.ExpressionProperty`) → `bool`

Test for equality.

__hash__ = `None`

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.ExpressionProperty) -> None`

Create a default ExpressionProperty with `key_only_filter = false` and `writer_side_filter_optimization = false`.

2. `__init__(self: rti.connextdds.ExpressionProperty, key_only_filter: bool, writer_side_filter_optimization: bool) -> None`

Create an `ExpressionProperty` with the provided `key_only_filter`, and `writer_side_filter_optimization`.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.ExpressionProperty, arg0: rti.connextdds.ExpressionProperty) -> bool`
Test for inequality.

property key_only_filter

The value for `key_only_filter`, indicating if the filter expression is based only on key fields. In this case, RTI Connex itself can cache the filtering results.

property writer_side_filter_optimization

The value for `writer_side_filter_optimization`, indicating if the filter implementation can cache the filtering result for the provided expression.

class rti.connextdds.ExtensibilityKind

Bases: `pybind11_object`

EXTENSIBLE = <ExtensibilityKind.EXTENSIBLE: 1>

class ExtensibilityKind

Bases: `pybind11_object`

Members:

FINAL : Final extensibility.

EXTENSIBLE : Extensible extensibility.

MUTABLE : Mutable extensibility.

EXTENSIBLE = <ExtensibilityKind.EXTENSIBLE: 1>

FINAL = <ExtensibilityKind.FINAL: 0>

MUTABLE = <ExtensibilityKind.MUTABLE: 2>

`__eq__(self: object, other: object) -> bool`

`__getstate__(self: object) -> int`

`__hash__(self: object) -> int`

`__index__(self: rti.connextdds.ExtensibilityKind.ExtensibilityKind) -> int`

`__init__(self: rti.connextdds.ExtensibilityKind.ExtensibilityKind, value: int) -> None`

`__int__(self: rti.connextdds.ExtensibilityKind.ExtensibilityKind) -> int`


```

__members__ = {'EXTENSIBLE': <ExtensibilityKind.EXTENSIBLE: 1>,
'FINAL': <ExtensibilityKind.FINAL: 0>, 'MUTABLE':
<ExtensibilityKind.MUTABLE: 2>}

__module__ = 'rti.connextdds'

__ne__ (self: object, other: object) → bool

__repr__ (self: object) → str

__setstate__ (self: rti.connextdds.ExtensibilityKind.ExtensibilityKind, state: int) → None

__str__ ()
    name(self: handle) -> str

property name

property value

FINAL = <ExtensibilityKind.FINAL: 0>

MUTABLE = <ExtensibilityKind.MUTABLE: 2>

__eq__ (self: rti.connextdds.ExtensibilityKind, arg0: rti.connextdds.ExtensibilityKind) → bool
    Apply operator to underlying enumerated values.

__ge__ (self: rti.connextdds.ExtensibilityKind, arg0: rti.connextdds.ExtensibilityKind) → bool
    Apply operator to underlying enumerated values.

__gt__ (self: rti.connextdds.ExtensibilityKind, arg0: rti.connextdds.ExtensibilityKind) → bool
    Apply operator to underlying enumerated values.

__hash__ = None

__init__ (*args, **kwargs)
    Overloaded function.

    1. __init__(self: rti.connextdds.ExtensibilityKind) -> None
        Initializes enum to 0.

    2. __init__(self: rti.connextdds.ExtensibilityKind, arg0: rti.connextdds.ExtensibilityKind.Ex-
        tensibilityKind) -> None
        Copy constructor.

__int__ (self: rti.connextdds.ExtensibilityKind) → rti.connextdds.ExtensibilityKind.ExtensibilityKind
    Int conversion.

__le__ (self: rti.connextdds.ExtensibilityKind, arg0: rti.connextdds.ExtensibilityKind) → bool
    Apply operator to underlying enumerated values.

__lt__ (self: rti.connextdds.ExtensibilityKind, arg0: rti.connextdds.ExtensibilityKind) → bool
    Apply operator to underlying enumerated values.

```

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ExtensibilityKind, *arg0*: rti.connexdds.ExtensibilityKind) → bool
Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.ExtensibilityKind) → str
String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.**Filter**

Bases: pybind11_object

SQL_FILTER_NAME = 'DDSSQL'

STRINGMATCH_FILTER_NAME = 'DDSSTRINGMATCH'

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.Filter, *expression*: str) -> None

Create a Filter with the specified expression.

2. **__init__**(*self*: rti.connexdds.Filter, *expression*: str, *parameters*: rti.connexdds.StringSeq) -> None

Create a Filter with the specified expression and parameters.

__iter__ (*self*: rti.connexdds.Filter) → Iterator

__module__ = 'rti.connexdds'

add_parameters (*self*: rti.connexdds.Filter, *parameter*: str) → rti.connexdds.Filter

Append a parameter to the Filter.

property expression

Get the Filter's expression.

property name

The name of the type of Filter.

property parameter_count

Get the number of parameters.

set_parameters (*self*: rti.connexdds.Filter, *arg0*: rti.connexdds.StringSeq) → None

Set the parameters for the Filter.

class rti.connexdds.**FilterSampleInfo**

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.FilterSampleInfo, *arg0*: rti.connexdds.FilterSampleInfo) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.FilterSampleInfo, arg0: rti.connexdds.FilterSampleInfo) → bool
Test for inequality.

property priority

Get a positive integer designating the relative priority of the sample, used to determine the transmission order of pending transmissions.

property related_sample_identity

The Identity of another sample related to this one.

class rti.connexdds.**Float128Seq**

Bases: pybind11_object

__add__ (self: rti.connexdds.Float128Seq, arg0: rti.connexdds.Float128Seq) → *rti.connexdds.Float128Seq*

__bool__ (self: rti.connexdds.Float128Seq) → bool
Check whether the list is nonempty

__contains__ (self: rti.connexdds.Float128Seq, x: rti.connexdds.LongDouble) → bool
Return true the container contains x

__delitem__ (*args, **kwargs)
Overloaded function.

1. **__delitem__**(self: rti.connexdds.Float128Seq, arg0: int) -> None

Delete the list elements at index i

2. **__delitem__**(self: rti.connexdds.Float128Seq, arg0: slice) -> None

Delete list elements using a slice object

__eq__ (self: rti.connexdds.Float128Seq, arg0: rti.connexdds.Float128Seq) → bool

__getitem__ (*args, **kwargs)
Overloaded function.

1. **__getitem__**(self: rti.connexdds.Float128Seq, s: slice) -> rti.connexdds.Float128Seq

Retrieve list elements using a slice object

2. **__getitem__**(self: rti.connexdds.Float128Seq, arg0: int) -> rti.connexdds.LongDouble

__hash__ = None

__iadd__ (self: rti.connexdds.Float128Seq, arg0: rti.connexdds.Float128Seq) → *rti.connexdds.Float128Seq*

__imul__ (*self*: rti.connexdds.Float128Seq, *arg0*: int) → *rti.connexdds.Float128Seq*

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: rti.connexdds.Float128Seq) -> None
2. **__init__**(*self*: rti.connexdds.Float128Seq, *arg0*: rti.connexdds.Float128Seq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.Float128Seq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.Float128Seq) → Iterator

__len__ (*self*: rti.connexdds.Float128Seq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.Float128Seq, *arg0*: int) → *rti.connexdds.Float128Seq*

__ne__ (*self*: rti.connexdds.Float128Seq, *arg0*: rti.connexdds.Float128Seq) → bool

__repr__ (*self*: rti.connexdds.Float128Seq) → str
Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.Float128Seq, *arg0*: int) → *rti.connexdds.Float128Seq*

__setitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.Float128Seq, *arg0*: int, *arg1*: rti.connexdds.LongDouble) -> None
2. **__setitem__**(*self*: rti.connexdds.Float128Seq, *arg0*: slice, *arg1*: rti.connexdds.Float128Seq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.Float128Seq, *x*: rti.connexdds.LongDouble) → None
Add an item to the end of the list

clear (*self*: rti.connexdds.Float128Seq) → None
Clear the contents

count (*self*: rti.connexdds.Float128Seq, *x*: rti.connexdds.LongDouble) → int
Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)
Overloaded function.

1. **extend**(*self*: rti.connexdds.Float128Seq, *L*: rti.connexdds.Float128Seq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.Float128Seq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.Float128Seq, *i*: int, *x*: rti.connexdds.LongDouble) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(*self*: rti.connexdds.Float128Seq) → rti.connexdds.LongDouble

Remove and return the last item

2. pop(*self*: rti.connexdds.Float128Seq, *i*: int) → rti.connexdds.LongDouble

Remove and return the item at index *i*

remove (*self*: rti.connexdds.Float128Seq, *x*: rti.connexdds.LongDouble) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.**Float128Type**

Bases: *DynamicType*

__eq__ (*self*: rti.connexdds.Float128Type, *arg0*: rti.connexdds.Float128Type) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.Float128Type) → None

Get the singleton for Float128Type

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.Float128Type, *arg0*: rti.connexdds.Float128Type) → bool

Test for inequality.

class rti.connexdds.**Float32Seq**

Bases: *pybind11_object*

__add__ (*self*: rti.connexdds.Float32Seq, *arg0*: rti.connexdds.Float32Seq) →
rti.connexdds.Float32Seq

__bool__ (*self*: rti.connexdds.Float32Seq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.Float32Seq, *x*: *float*) → bool

Return true the container contains *x*

__delitem__ (*args, **kwargs)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.Float32Seq, *arg0*: int) → None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.Float32Seq, *arg0*: slice) → None

Delete list elements using a slice object

`__eq__` (*self*: rti.connexdds.Float32Seq, *arg0*: rti.connexdds.Float32Seq) → bool

`__getitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__`(*self*: rti.connexdds.Float32Seq, *s*: slice) → rti.connexdds.Float32Seq

Retrieve list elements using a slice object

2. `__getitem__`(*self*: rti.connexdds.Float32Seq, *arg0*: int) → float

`__getstate__` (*self*: rti.connexdds.Float32Seq) → bytes

`__hash__` = None

`__iadd__` (*self*: rti.connexdds.Float32Seq, *arg0*: rti.connexdds.Float32Seq) →
rti.connexdds.Float32Seq

`__imul__` (*self*: rti.connexdds.Float32Seq, *arg0*: int) → *rti.connexdds.Float32Seq*

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__`(*self*: rti.connexdds.Float32Seq, *arg0*: buffer) → None
2. `__init__`(*self*: rti.connexdds.Float32Seq) → None
3. `__init__`(*self*: rti.connexdds.Float32Seq, *arg0*: rti.connexdds.Float32Seq) → None

Copy constructor

4. `__init__`(*self*: rti.connexdds.Float32Seq, *arg0*: Iterable) → None
5. `__init__`(*self*: rti.connexdds.Float32Seq, *arg0*: int) → None

`__iter__` (*self*: rti.connexdds.Float32Seq) → Iterator

`__len__` (*self*: rti.connexdds.Float32Seq) → int

`__module__` = 'rti.connexdds'

`__mul__` (*self*: rti.connexdds.Float32Seq, *arg0*: int) → *rti.connexdds.Float32Seq*

`__ne__` (*self*: rti.connexdds.Float32Seq, *arg0*: rti.connexdds.Float32Seq) → bool

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` = <capsule object NULL>

`__repr__` (*self*: rti.connexdds.Float32Seq) → str

Return the canonical string representation of this list.

`__rmul__` (*self*: rti.connexdds.Float32Seq, *arg0*: int) → *rti.connexdds.Float32Seq*

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.Float32Seq, arg0: int, arg1: float) -> None`
2. `__setitem__(self: rti.connextdds.Float32Seq, arg0: slice, arg1: rti.connextdds.Float32Seq) -> None`

Assign list elements using a slice object

__setstate__ (self: rti.connextdds.Float32Seq, arg0: bytes) → None

append (self: rti.connextdds.Float32Seq, x: float) → None

Add an item to the end of the list

clear (self: rti.connextdds.Float32Seq) → None

Clear the contents

count (self: rti.connextdds.Float32Seq, x: float) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connextdds.Float32Seq, L: rti.connextdds.Float32Seq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.Float32Seq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.Float32Seq, i: int, x: float) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.Float32Seq) -> float`

Remove and return the last item

2. `pop(self: rti.connextdds.Float32Seq, i: int) -> float`

Remove and return the item at index i

remove (self: rti.connextdds.Float32Seq, x: float) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

resize (self: rti.connextdds.Float32Seq, arg0: int) → None

resizes the vector to the given size

class rti.connextdds.Float32Type

Bases: *DynamicType*

`__eq__` (*self*: rti.connexdds.Float32Type, *arg0*: rti.connexdds.Float32Type) → bool

Test for equality.

`__hash__` = None

`__init__` (*self*: rti.connexdds.Float32Type) → None

Get the singleton for Float32Type

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.Float32Type, *arg0*: rti.connexdds.Float32Type) → bool

Test for inequality.

class rti.connexdds.Float64Seq

Bases: pybind11_object

`__add__` (*self*: rti.connexdds.Float64Seq, *arg0*: rti.connexdds.Float64Seq) →

rti.connexdds.Float64Seq

`__bool__` (*self*: rti.connexdds.Float64Seq) → bool

Check whether the list is nonempty

`__contains__` (*self*: rti.connexdds.Float64Seq, *x*: float) → bool

Return true the container contains *x*

`__delitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__delitem__`(*self*: rti.connexdds.Float64Seq, *arg0*: int) -> None

Delete the list elements at index *i*

2. `__delitem__`(*self*: rti.connexdds.Float64Seq, *arg0*: slice) -> None

Delete list elements using a slice object

`__eq__` (*self*: rti.connexdds.Float64Seq, *arg0*: rti.connexdds.Float64Seq) → bool

`__getitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__`(*self*: rti.connexdds.Float64Seq, *s*: slice) -> rti.connexdds.Float64Seq

Retrieve list elements using a slice object

2. `__getitem__`(*self*: rti.connexdds.Float64Seq, *arg0*: int) -> float

`__getstate__` (*self*: rti.connexdds.Float64Seq) → bytes

`__hash__` = None

`__iadd__` (*self*: rti.connexdds.Float64Seq, *arg0*: rti.connexdds.Float64Seq) →

rti.connexdds.Float64Seq

__imul__ (*self*: rti.connexdds.Float64Seq, *arg0*: int) → rti.connexdds.Float64Seq

__init__ (*args, **kwargs)
Overloaded function.

1. **__init__**(self: rti.connexdds.Float64Seq, arg0: buffer) -> None
2. **__init__**(self: rti.connexdds.Float64Seq) -> None
3. **__init__**(self: rti.connexdds.Float64Seq, arg0: rti.connexdds.Float64Seq) -> None

Copy constructor

4. **__init__**(self: rti.connexdds.Float64Seq, arg0: Iterable) -> None
5. **__init__**(self: rti.connexdds.Float64Seq, arg0: int) -> None

__iter__ (*self*: rti.connexdds.Float64Seq) → Iterator

__len__ (*self*: rti.connexdds.Float64Seq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.Float64Seq, *arg0*: int) → rti.connexdds.Float64Seq

__ne__ (*self*: rti.connexdds.Float64Seq, *arg0*: rti.connexdds.Float64Seq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (*self*: rti.connexdds.Float64Seq) → str
Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.Float64Seq, *arg0*: int) → rti.connexdds.Float64Seq

__setitem__ (*args, **kwargs)
Overloaded function.

1. **__setitem__**(self: rti.connexdds.Float64Seq, arg0: int, arg1: float) -> None
2. **__setitem__**(self: rti.connexdds.Float64Seq, arg0: slice, arg1: rti.connexdds.Float64Seq) -> None

Assign list elements using a slice object

__setstate__ (*self*: rti.connexdds.Float64Seq, *arg0*: bytes) → None

append (*self*: rti.connexdds.Float64Seq, *x*: float) → None
Add an item to the end of the list

clear (*self*: rti.connexdds.Float64Seq) → None
Clear the contents

count (*self*: rti.connexdds.Float64Seq, *x*: float) → int
Return the number of times *x* appears in the list

extend (*args, **kwargs)

Overloaded function.

1. extend(self: rti.connexdds.Float64Seq, L: rti.connexdds.Float64Seq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connexdds.Float64Seq, L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (self: rti.connexdds.Float64Seq, i: int, x: float) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connexdds.Float64Seq) -> float

Remove and return the last item

2. pop(self: rti.connexdds.Float64Seq, i: int) -> float

Remove and return the item at index i

remove (self: rti.connexdds.Float64Seq, x: float) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

resize (self: rti.connexdds.Float64Seq, arg0: int) → None

resizes the vector to the given size

class rti.connexdds.**Float64Type**

Bases: *DynamicType*

__eq__ (self: rti.connexdds.Float64Type, arg0: rti.connexdds.Float64Type) → bool

Test for equality.

__hash__ = None

__init__ (self: rti.connexdds.Float64Type) → None

Get the singleton for Float64Type

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.Float64Type, arg0: rti.connexdds.Float64Type) → bool

Test for inequality.

rti.connexdds.**FloatSeq**

alias of *Float32Seq*

rti.connexdds.**FloatType**

alias of *Float32Type*

class rti.connexdds.**FlowController**

Bases: *pybind11_object*

DEFAULT_NAME = 'DDS_DEFAULT_FLOW_CONTROLLER_NAME'

FIXED_RATE_NAME = 'DDS_FIXED_RATE_FLOW_CONTROLLER_NAME'

ON_DEMAND_NAME = 'DDS_ON_DEMAND_FLOW_CONTROLLER_NAME'

__eq__ (*self*: rti.connexdds.FlowController, *arg0*: rti.connexdds.FlowController) → bool
Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.FlowController, *participant*: rti.connexdds.DomainParticipant, *name*: str, *token_bucket*: rti.connexdds.FlowControllerProperty = *FlowControllerProperty*()) → None
Creates a FlowController with specific properties.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.FlowController, *arg0*: rti.connexdds.FlowController) → bool
Test for inequality.

close (*self*: rti.connexdds.FlowController) → None
Manually destroys this object.

property closed

Whether this FlowController has been closed.

property name

The name of this FlowController.

property participant

The participant of this FlowController.

property property

The configuration of this FlowController.

retain (*self*: rti.connexdds.FlowController) → None
Disables the automatic destruction of this object.

trigger_flow (*self*: rti.connexdds.FlowController) → None
Provides an external way to trigger a FlowController.

class rti.connexdds.**FlowControllerProperty**

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.FlowControllerProperty, *arg0*: rti.connexdds.FlowControllerProperty) → bool
Test for equality.

__hash__ = None

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.FlowControllerProperty) -> None`

Creates a `FlowControllerProperty` with earliest-deadline-first scheduling policy and default token-bucket configuration.

2. `__init__(self: rti.connexdds.FlowControllerProperty, scheduling_policy: rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy, token_bucket: rti.connexdds.FlowControllerTokenBucketProperty = FlowControllerTokenBucketProperty()) -> None`

Creates a `FlowControllerProperty` with the specified scheduling policy and token-bucket configuration.

`__module__` = 'rti.connexdds'

`__ne__` (self: rti.connexdds.FlowControllerProperty, arg0: rti.connexdds.FlowControllerProperty) → bool

Test for inequality.

property `scheduling_policy`

The scheduling policy.

property `token_bucket`

The token-bucket configuration

class `rti.connexdds.FlowControllerSchedulingPolicy`

Bases: `pybind11_object`

EARLIEST_DEADLINE_FIRST =

`<FlowControllerSchedulingPolicy.EARLIEST_DEADLINE_FIRST: 1>`

class `FlowControllerSchedulingPolicy`

Bases: `pybind11_object`

Members:

`ROUND_ROBIN` : Indicates to flow control in a round-robin fashion.

Whenever tokens become available, the flow controller distributes the tokens uniformly across all of its (non-empty) destination queues. No destinations are prioritized. Instead, all destinations are treated equally and are serviced in a round-robin fashion.

`EARLIEST_DEADLINE_FIRST` : Indicates to flow control in an earliest-deadline-first fashion.

A sample's deadline is determined by the time it was written plus the latency budget of the `DataWriter` at the time of the write call (as specified in the `LatencyBudget`). The relative priority of a flow controller's destination queue is determined by the earliest deadline across all samples it contains.

When tokens become available, the `FlowController` distributes tokens to the destination queues in order of their deadline priority. In other words, the queue containing the sample with the earliest

deadline is serviced first. The number of tokens granted equals the number of tokens required to send the first sample in the queue. Note that the priority of a queue may change as samples are sent (i.e. removed from the queue). If a sample must be sent to multiple destinations or two samples have an equal deadline value, the corresponding destination queues are serviced in a round-robin fashion.

Hence, under the default `LatencyBudget::duration` setting, an `EDF_FLOW_CONTROLLER_SCHED_POLICY` FlowController preserves the order in which the user calls `DataWriter.write()` across the DataWriters associated with the flow controller.

Since the `LatencyBudget` is mutable, a sample written second may contain an earlier deadline than the sample written first if the `LatencyBudget.duration` value is sufficiently decreased in between writing the two samples. In that case, if the first sample is not yet written (still in queue waiting for its turn), it inherits the priority corresponding to the (earlier) deadline from the second sample.

In other words, the priority of a destination queue is always determined by the earliest deadline among all samples contained in the queue. This priority inheritance approach is required in order to both honor the updated `LatencyBudget.duration` and adhere to the DataWriter in-order data delivery guarantee.

[default] for DataWriter

HIGHEST_PRIORITY_FIRST : Indicates to flow control in a highest-priority-first fashion.

Determines the next destination queue to service as determined by the publication priority of the DataWriter, channel of multi-channel DataWriter, or individual sample.

The relative priority of a flow controller's destination queue is determined by the highest publication priority of all samples it contains.

When tokens become available, the FlowController distributes tokens to the destination queues in order of their publication priority. In other words, the queue containing the sample with the highest publication priority is serviced first. The number of tokens granted equals the number of tokens required to send the first sample in the queue. Note that the priority of a queue may change as samples are sent (i.e. removed from the queue). If a sample must be sent to multiple destinations or two samples have an equal publication priority, the corresponding destination queues are serviced in a round-robin fashion.

This priority inheritance approach is required in order to both honor the designated publication priority and adhere to the DataWriter in-order data delivery guarantee.

EARLIEST_DEADLINE_FIRST =
<FlowControllerSchedulingPolicy.EARLIEST_DEADLINE_FIRST: 1>

HIGHEST_PRIORITY_FIRST =
<FlowControllerSchedulingPolicy.HIGHEST_PRIORITY_FIRST: 2>

ROUND_ROBIN = <FlowControllerSchedulingPolicy.ROUND_ROBIN: 0>

`__eq__` (*self: object, other: object*) → bool

`__getstate__` (*self: object*) → int

```

__hash__(self: object) → int

__index__(self:
    rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy)
    → int

__init__(self:
    rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy,
    value: int) → None

__int__(self:
    rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy) →
int

__members__ = {'EARLIEST_DEADLINE_FIRST':
<FlowControllerSchedulingPolicy.EARLIEST_DEADLINE_FIRST: 1>,
'HIGHEST_PRIORITY_FIRST':
<FlowControllerSchedulingPolicy.HIGHEST_PRIORITY_FIRST: 2>,
'ROUND_ROBIN': <FlowControllerSchedulingPolicy.ROUND_ROBIN: 0>}

__module__ = 'rti.connexdds'

__ne__(self: object, other: object) → bool

__repr__(self: object) → str

__setstate__(self: rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedul-
ingPolicy, state: int) →
None

__str__()
    name(self: handle) -> str

property name

property value

HIGHEST_PRIORITY_FIRST =
<FlowControllerSchedulingPolicy.HIGHEST_PRIORITY_FIRST: 2>

ROUND_ROBIN = <FlowControllerSchedulingPolicy.ROUND_ROBIN: 0>

__eq__(self: rti.connexdds.FlowControllerSchedulingPolicy, arg0:
    rti.connexdds.FlowControllerSchedulingPolicy) → bool
    Apply operator to underlying enumerated values.

__ge__(self: rti.connexdds.FlowControllerSchedulingPolicy, arg0:
    rti.connexdds.FlowControllerSchedulingPolicy) → bool
    Apply operator to underlying enumerated values.

```

__gt__ (*self*: rti.connexdds.FlowControllerSchedulingPolicy, *arg0*: rti.connexdds.FlowControllerSchedulingPolicy) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.FlowControllerSchedulingPolicy) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.FlowControllerSchedulingPolicy, *arg0*: rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.FlowControllerSchedulingPolicy) → *rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy*

Int conversion.

__le__ (*self*: rti.connexdds.FlowControllerSchedulingPolicy, *arg0*: rti.connexdds.FlowControllerSchedulingPolicy) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.FlowControllerSchedulingPolicy, *arg0*: rti.connexdds.FlowControllerSchedulingPolicy) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.FlowControllerSchedulingPolicy, *arg0*: rti.connexdds.FlowControllerSchedulingPolicy) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.FlowControllerSchedulingPolicy) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.**FlowControllerTokenBucketProperty**

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.FlowControllerTokenBucketProperty, *arg0*: rti.connexdds.FlowControllerTokenBucketProperty) → bool

Test for equality.

__hash__ = None

```
__init__ (self: rti.connexdds.FlowControllerTokenBucketProperty, max_tokens: int =
    LENGTH_UNLIMITED, tokens_added_per_period: int = LENGTH_UNLIMITED,
    tokens_leaked_per_period: int = 0, period: rti.connexdds.Duration = Duration(1, 0),
    bytes_per_token: int = LENGTH_UNLIMITED) → None
```

Initializes the properties.

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.FlowControllerTokenBucketProperty, arg0:
    rti.connexdds.FlowControllerTokenBucketProperty) → bool
```

Test for inequality.

```
property bytes_per_token
```

The maximum number of bytes allowed to send for each token available.

```
property max_tokens
```

The maximum number of tokens that can accumulate in the token bucket.

```
property period
```

The period for adding tokens to and removing tokens from the bucket.

```
property tokens_added_per_period
```

The number of tokens added to the token bucket per specified period.

```
property tokens_leaked_per_period
```

The number of tokens removed from the token bucket per specified period.

```
class rti.connexdds.GenerationCount
```

```
Bases: pybind11_object
```

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connexdds.GenerationCount) -> None`

Create a default GenerationCount object.

2. `__init__(self: rti.connexdds.GenerationCount, disposed_count: int, no_writers_count: int) -> None`

Create a GenerationCount object with the provided `disposed_count` and `no_writers` count.

```
__module__ = 'rti.connexdds'
```

```
property disposed
```

Get the disposed generation count.

```
property no_writers
```

Get the `no_writers` generation count.

```
class rti.connexdds.GroupData
```

```
Bases: pybind11_object
```


__eq__ (*self*: rti.connexdds.GroupData, *arg0*: rti.connexdds.GroupData) → bool
Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.GroupData) -> None

Create a GroupData instance.

2. **__init__**(*self*: rti.connexdds.GroupData, *bytes*: rti.connexdds.Uint8Seq) -> None

Create a GroupData instance with a sequence of bytes.

__iter__ (*self*: rti.connexdds.GroupData) → Iterator

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.GroupData, *arg0*: rti.connexdds.GroupData) → bool
Test for inequality.

property value

The byte sequence of this GroupData.

class rti.connexdds.GuardCondition

Bases: *ICondition*

__eq__ (*self*: rti.connexdds.GuardCondition, *arg0*: rti.connexdds.GuardCondition) → bool
Compare GuardCondition objects for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.GuardCondition) -> None

Create a GuardCondition in an untriggered state.

2. **__init__**(*self*: rti.connexdds.GuardCondition, *condition*: rti.connexdds.ICondition) -> None

Create a GuardCondition from a Condition.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.GuardCondition, *arg0*: rti.connexdds.GuardCondition) → bool
Compare GuardCondition objects for inequality.

reset_handler (*self*: rti.connexdds.GuardCondition) → None
Resets the handler for this GuardCondition.

set_handler (*self*: rti.connextdds.GuardCondition, *func*: Callable[[rti.connextdds.GuardCondition], None]) → None

Set a handler function receiving the Condition as its only argument.

set_handler_no_args (*self*: rti.connextdds.GuardCondition, *func*: Callable[[], None]) → None

Set a handler function with no arguments.

property trigger_value

Get/set the trigger value for this GuardCondition

class rti.connextdds.Guid

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.Guid, *arg0*: rti.connextdds.Guid) → bool

Compare Guids for equality.

__ge__ (*self*: rti.connextdds.Guid, *arg0*: rti.connextdds.Guid) → bool

Compare Guids.

__getitem__ (*self*: rti.connextdds.Guid, *arg0*: int) → int

__gt__ (*self*: rti.connextdds.Guid, *arg0*: rti.connextdds.Guid) → bool

Compare Guids.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.Guid) -> None

Create a default Guid (equivalent to unknown Guid).

2. **__init__**(*self*: rti.connextdds.Guid, builtin_topic_key: rti.connextdds.BuiltinTopicKey) -> None

Create a Guid from a BuiltinTopicKey.

__le__ (*self*: rti.connextdds.Guid, *arg0*: rti.connextdds.Guid) → bool

Compare Guids.

__len__ (*self*: rti.connextdds.Guid) → int

__lt__ (*self*: rti.connextdds.Guid, *arg0*: rti.connextdds.Guid) → bool

Compare Guids.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.Guid, *arg0*: rti.connextdds.Guid) → bool

Compare Guids for inequality.

__setitem__ (*self*: rti.connextdds.Guid, *arg0*: int, *arg1*: int) → int

`__str__` (*self*: rti.connexdds.Guid) → str

`automatic` = <rti.connexdds.Guid object>

`unknown` = <rti.connexdds.Guid object>

class rti.connexdds.History

Bases: pybind11_object

`__eq__` (*self*: rti.connexdds.History, *arg0*: rti.connexdds.History) → bool

Test for equality.

`__hash__` = None

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.History) -> None`

Creates a policy that keeps the last sample only.

2. `__init__(self: rti.connexdds.History, kind: rti.connexdds.HistoryKind, depth: int = 1) -> None`

Creates a policy with a specific history kind and optionally a history depth.

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.History, *arg0*: rti.connexdds.History) → bool

Test for inequality.

property depth

The history kind.

`keep_all` = <rti.connexdds.History object>

static `keep_last` (*depth*: int) → rti.connexdds.History

Creates a History with HistoryKind.KEEP_LAST and the specified depth.

property kind

The history kind.

class rti.connexdds.HistoryKind

Bases: pybind11_object

class HistoryKind

Bases: pybind11_object

Members:

KEEP_LAST : [default] Keep the last depth samples.

On the publishing side, RTI Connex will only attempt to keep the most recent depth samples of each instance of data (identified by its key) managed by the DataWriter. Invalid samples representing a disposal or unregistration of an instance do not count towards the history depth.

On the subscribing side, the DataReader will only attempt to keep the most recent depth samples received for each instance (identified by its key) until the application takes them via the DataReader's take() operation.

Invalid samples representing a disposal or unregistration of an instance do count towards the history depth and may therefore replace a value of the instance currently being stored in the reader queue.

KEEP_ALL : Keep all the samples.

On the publishing side, RTI Connex will attempt to keep all samples (representing each value written) of each instance of data (identified by its key) managed by the DataWriter until they can be delivered to all subscribers.

On the subscribing side, RTI Connex will attempt to keep all samples of each instance of data (identified by its key) managed by the DataReader. These samples are kept until the application takes them from RTI Connex via the take() operation.

KEEP_ALL = <HistoryKind.KEEP_ALL: 1>

KEEP_LAST = <HistoryKind.KEEP_LAST: 0>

__eq__ (self: object, other: object) → bool

__getstate__ (self: object) → int

__hash__ (self: object) → int

__index__ (self: rti.connexdds.HistoryKind.HistoryKind) → int

__init__ (self: rti.connexdds.HistoryKind.HistoryKind, value: int) → None

__int__ (self: rti.connexdds.HistoryKind.HistoryKind) → int

__members__ = {'KEEP_ALL': <HistoryKind.KEEP_ALL: 1>, 'KEEP_LAST': <HistoryKind.KEEP_LAST: 0>}

__module__ = 'rti.connexdds'

__ne__ (self: object, other: object) → bool

__repr__ (self: object) → str

__setstate__ (self: rti.connexdds.HistoryKind.HistoryKind, state: int) → None

__str__ ()

name(self: handle) -> str

property name

property value

KEEP_ALL = <HistoryKind.KEEP_ALL: 1>

KEEP_LAST = <HistoryKind.KEEP_LAST: 0>

__eq__ (*self*: rti.connexdds.HistoryKind, *arg0*: rti.connexdds.HistoryKind) → bool
Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.HistoryKind, *arg0*: rti.connexdds.HistoryKind) → bool
Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.HistoryKind, *arg0*: rti.connexdds.HistoryKind) → bool
Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)
Overloaded function.

- __init__**(*self*: rti.connexdds.HistoryKind) -> None
Initializes enum to 0.
- __init__**(*self*: rti.connexdds.HistoryKind, *arg0*: rti.connexdds.HistoryKind.HistoryKind) -> None
Copy constructor.

__int__ (*self*: rti.connexdds.HistoryKind) → *rti.connexdds.HistoryKind.HistoryKind*
Int conversion.

__le__ (*self*: rti.connexdds.HistoryKind, *arg0*: rti.connexdds.HistoryKind) → bool
Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.HistoryKind, *arg0*: rti.connexdds.HistoryKind) → bool
Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.HistoryKind, *arg0*: rti.connexdds.HistoryKind) → bool
Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.HistoryKind) → str
String conversion.

property underlying
Retrieves the actual enumerated value.

class rti.connexdds.IAnyDataReader
Bases: pybind11_object

__eq__ (*self*: rti.connexdds.IAnyDataReader, *arg0*: rti.connexdds.IAnyDataReader) → bool
Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.IAnyDataReader, *arg0*: rti.connexdds.IAnyDataReader) → bool
Test for inequality.

close (*self*: rti.connexdds.IAnyDataReader) → None
Close this DataReader.

property qos

The QoS for this AnyDataReader.

This property's getter returns a deep copy.

retain (*self*: rti.connexdds.IAnyDataReader) → None
Retain this DataReader.

property subscriber

The Publisher for this AnyDataReader.

property topic_name

The Topic name for this AnyDataReader.

property type_name

The type name for this AnyDataReader.

class rti.connexdds.IAnyDataReaderSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.IAnyDataReaderSeq, *arg0*: rti.connexdds.IAnyDataReaderSeq) → *rti.connexdds.IAnyDataReaderSeq*

__bool__ (*self*: rti.connexdds.IAnyDataReaderSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.IAnyDataReaderSeq, *x*: rti.connexdds.IAnyDataReader) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.IAnyDataReaderSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.IAnyDataReaderSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.IAnyDataReaderSeq, *arg0*: rti.connexdds.IAnyDataReaderSeq) → bool

__getitem__ (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connexdds.IAnyDataReaderSeq, s: slice) -> rti.connexdds.IAnyDataReaderSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.IAnyDataReaderSeq, arg0: int) -> rti.connexdds.IAnyDataReader`

__hash__ = None

__iadd__ (self: rti.connexdds.IAnyDataReaderSeq, arg0: rti.connexdds.IAnyDataReaderSeq) → *rti.connexdds.IAnyDataReaderSeq*

__imul__ (self: rti.connexdds.IAnyDataReaderSeq, arg0: int) → *rti.connexdds.IAnyDataReaderSeq*

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.IAnyDataReaderSeq) -> None`
2. `__init__(self: rti.connexdds.IAnyDataReaderSeq, arg0: rti.connexdds.IAnyDataReaderSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.IAnyDataReaderSeq, arg0: Iterable) -> None`

__iter__ (self: rti.connexdds.IAnyDataReaderSeq) → Iterator

__len__ (self: rti.connexdds.IAnyDataReaderSeq) → int

__module__ = 'rti.connexdds'

__mul__ (self: rti.connexdds.IAnyDataReaderSeq, arg0: int) → *rti.connexdds.IAnyDataReaderSeq*

__ne__ (self: rti.connexdds.IAnyDataReaderSeq, arg0: rti.connexdds.IAnyDataReaderSeq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (self: rti.connexdds.IAnyDataReaderSeq) → str

Return the canonical string representation of this list.

__rmul__ (self: rti.connexdds.IAnyDataReaderSeq, arg0: int) → *rti.connexdds.IAnyDataReaderSeq*

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.IAnyDataReaderSeq, arg0: int, arg1: rti.connextdds.IAnyDataReader) -> None`
2. `__setitem__(self: rti.connextdds.IAnyDataReaderSeq, arg0: slice, arg1: rti.connextdds.IAnyDataReaderSeq) -> None`

Assign list elements using a slice object

append (*self*: rti.connextdds.IAnyDataReaderSeq, *x*: rti.connextdds.IAnyDataReader) → None

Add an item to the end of the list

clear (*self*: rti.connextdds.IAnyDataReaderSeq) → None

Clear the contents

count (*self*: rti.connextdds.IAnyDataReaderSeq, *x*: rti.connextdds.IAnyDataReader) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. `extend(self: rti.connextdds.IAnyDataReaderSeq, L: rti.connextdds.IAnyDataReaderSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.IAnyDataReaderSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.IAnyDataReaderSeq, *i*: int, *x*: rti.connextdds.IAnyDataReader) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connextdds.IAnyDataReaderSeq) -> rti.connextdds.IAnyDataReader`

Remove and return the last item

2. `pop(self: rti.connextdds.IAnyDataReaderSeq, i: int) -> rti.connextdds.IAnyDataReader`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.IAnyDataReaderSeq, *x*: rti.connextdds.IAnyDataReader) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class `rti.connextdds.IAnyDataWriter`

Bases: `pybind11_object`

__eq__ (*self*: rti.connextdds.IAnyDataWriter, *arg0*: rti.connextdds.IAnyDataWriter) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.IAnyDataWriter, *arg0*: rti.connexdds.IAnyDataWriter) → bool
Test for inequality.

close (*self*: rti.connexdds.IAnyDataWriter) → None
Close this DataWriter.

property publisher
The Publisher for this AnyDataWriter.

property qos
The QoS for this AnyDataWriter.
This property's getter returns a deep copy.

retain (*self*: rti.connexdds.IAnyDataWriter) → None
Retain this DataWriter.

property topic_name
The Topic name for this AnyDataWriter.

property type_name
The type name for this AnyDataWriter.

wait_for_acknowledgments (*self*: rti.connexdds.IAnyDataWriter, *timeout*: rti.connexdds.Duration) → None
Wait for acknowledgments from subscribers.

class rti.connexdds.IAnyDataWriterSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.IAnyDataWriterSeq, *arg0*: rti.connexdds.IAnyDataWriterSeq) → rti.connexdds.IAnyDataWriterSeq

__bool__ (*self*: rti.connexdds.IAnyDataWriterSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.IAnyDataWriterSeq, *x*: rti.connexdds.IAnyDataWriter) → bool
Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.IAnyDataWriterSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.IAnyDataWriterSeq, *arg0*: slice) -> None

Delete list elements using a slice object

`__eq__` (*self*: rti.connexdds.IAnyDataWriterSeq, *arg0*: rti.connexdds.IAnyDataWriterSeq) → bool

`__getitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__(self: rti.connexdds.IAnyDataWriterSeq, s: slice) -> rti.connexdds.IAnyDataWriterSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.IAnyDataWriterSeq, arg0: int) -> rti.connexdds.IAnyDataWriter`

`__hash__` = None

`__iadd__` (*self*: rti.connexdds.IAnyDataWriterSeq, *arg0*: rti.connexdds.IAnyDataWriterSeq) → *rti.connexdds.IAnyDataWriterSeq*

`__imul__` (*self*: rti.connexdds.IAnyDataWriterSeq, *arg0*: int) → *rti.connexdds.IAnyDataWriterSeq*

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.IAnyDataWriterSeq) -> None`
2. `__init__(self: rti.connexdds.IAnyDataWriterSeq, arg0: rti.connexdds.IAnyDataWriterSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.IAnyDataWriterSeq, arg0: Iterable) -> None`

`__iter__` (*self*: rti.connexdds.IAnyDataWriterSeq) → Iterator

`__len__` (*self*: rti.connexdds.IAnyDataWriterSeq) → int

`__module__` = 'rti.connexdds'

`__mul__` (*self*: rti.connexdds.IAnyDataWriterSeq, *arg0*: int) → *rti.connexdds.IAnyDataWriterSeq*

`__ne__` (*self*: rti.connexdds.IAnyDataWriterSeq, *arg0*: rti.connexdds.IAnyDataWriterSeq) → bool

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` = <capsule object NULL>

`__repr__` (*self*: rti.connexdds.IAnyDataWriterSeq) → str

Return the canonical string representation of this list.

`__rmul__` (*self*: rti.connexdds.IAnyDataWriterSeq, *arg0*: int) → *rti.connexdds.IAnyDataWriterSeq*

`__setitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__setitem__(self: rti.connexdds.IAnyDataWriterSeq, arg0: int, arg1: rti.connexdds.IAnyDataWriter) -> None`

2. `__setitem__(self: rti.connexdds.IAnyDataWriterSeq, arg0: slice, arg1: rti.connexdds.IAnyDataWriterSeq) -> None`

Assign list elements using a slice object

append (*self*: rti.connexdds.IAnyDataWriterSeq, *x*: rti.connexdds.IAnyDataWriter) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.IAnyDataWriterSeq) → None

Clear the contents

count (*self*: rti.connexdds.IAnyDataWriterSeq, *x*: rti.connexdds.IAnyDataWriter) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. `extend(self: rti.connexdds.IAnyDataWriterSeq, L: rti.connexdds.IAnyDataWriterSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.IAnyDataWriterSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.IAnyDataWriterSeq, *i*: int, *x*: rti.connexdds.IAnyDataWriter) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connexdds.IAnyDataWriterSeq) -> rti.connexdds.IAnyDataWriter`

Remove and return the last item

2. `pop(self: rti.connexdds.IAnyDataWriterSeq, i: int) -> rti.connexdds.IAnyDataWriter`

Remove and return the item at index *i*

remove (*self*: rti.connexdds.IAnyDataWriterSeq, *x*: rti.connexdds.IAnyDataWriter) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class `rti.connexdds.IAnyTopic`

Bases: `pybind11_object`

__eq__ (*self*: rti.connexdds.IAnyTopic, *arg0*: rti.connexdds.IAnyTopic) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.IAnyTopic, *arg0*: rti.connexdds.IAnyTopic) → bool
Test for inequality.

close (*self*: rti.connexdds.IAnyTopic) → None
Close this Topic.

property name
The Topic name for this AnyTopic.

property qos
The QoS for this AnyTopic.
This property's getter returns a deep copy.

property type_name
The type name for this AnyTopic.

class rti.connexdds.IAnyTopicSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.IAnyTopicSeq, *arg0*: rti.connexdds.IAnyTopicSeq) → *rti.connexdds.IAnyTopicSeq*

__bool__ (*self*: rti.connexdds.IAnyTopicSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.IAnyTopicSeq, *x*: rti.connexdds.IAnyTopic) → bool
Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.IAnyTopicSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.IAnyTopicSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.IAnyTopicSeq, *arg0*: rti.connexdds.IAnyTopicSeq) → bool

__getitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.IAnyTopicSeq, *s*: slice) -> rti.connexdds.IAnyTopicSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.IAnyTopicSeq, *arg0*: int) -> rti.connexdds.IAnyTopic

__hash__ = None

__iadd__ (*self*: rti.connexdds.IAnyTopicSeq, *arg0*: rti.connexdds.IAnyTopicSeq) → *rti.connexdds.IAnyTopicSeq*

`__imul__` (*self*: rti.connexdds.IAnyTopicSeq, *arg0*: int) → rti.connexdds.IAnyTopicSeq

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.IAnyTopicSeq)` -> None
2. `__init__(self: rti.connexdds.IAnyTopicSeq, arg0: rti.connexdds.IAnyTopicSeq)` -> None

Copy constructor

3. `__init__(self: rti.connexdds.IAnyTopicSeq, arg0: Iterable)` -> None

`__iter__` (*self*: rti.connexdds.IAnyTopicSeq) → Iterator

`__len__` (*self*: rti.connexdds.IAnyTopicSeq) → int

`__module__` = 'rti.connexdds'

`__mul__` (*self*: rti.connexdds.IAnyTopicSeq, *arg0*: int) → rti.connexdds.IAnyTopicSeq

`__ne__` (*self*: rti.connexdds.IAnyTopicSeq, *arg0*: rti.connexdds.IAnyTopicSeq) → bool

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` = <capsule object NULL>

`__repr__` (*self*: rti.connexdds.IAnyTopicSeq) → str

Return the canonical string representation of this list.

`__rmul__` (*self*: rti.connexdds.IAnyTopicSeq, *arg0*: int) → rti.connexdds.IAnyTopicSeq

`__setitem__` (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connexdds.IAnyTopicSeq, arg0: int, arg1: rti.connexdds.IAnyTopic)`
-> None
2. `__setitem__(self: rti.connexdds.IAnyTopicSeq, arg0: slice, arg1: rti.connexdds.IAnyTopicSeq)`
-> None

Assign list elements using a slice object

`append` (*self*: rti.connexdds.IAnyTopicSeq, *x*: rti.connexdds.IAnyTopic) → None

Add an item to the end of the list

`clear` (*self*: rti.connexdds.IAnyTopicSeq) → None

Clear the contents

`count` (*self*: rti.connexdds.IAnyTopicSeq, *x*: rti.connexdds.IAnyTopic) → int

Return the number of times *x* appears in the list

`extend` (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connexdds.IAnyTopicSeq, L: rti.connexdds.IAnyTopicSeq)` -> None

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.IAnyTopicSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.IAnyTopicSeq, *i*: int, *x*: rti.connextdds.IAnyTopic) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.IAnyTopicSeq) -> rti.connextdds.IAnyTopic`

Remove and return the last item

2. `pop(self: rti.connextdds.IAnyTopicSeq, i: int) -> rti.connextdds.IAnyTopic`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.IAnyTopicSeq, *x*: rti.connextdds.IAnyTopic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class `rti.connextdds.ICondition`

Bases: `pybind11_object`

__eq__ (*self*: rti.connextdds.ICondition, *arg0*: rti.connextdds.ICondition) → bool

Compare Condition objects for equality.

__hash__ = None

__init__ (*args, **kwargs)

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.ICondition, *arg0*: rti.connextdds.ICondition) → bool

Compare Condition objects for inequality.

dispatch (*self*: rti.connextdds.ICondition) → None

Dispatches the functions registered with the condition.

property trigger_value

The trigger value of the condition.

class `rti.connextdds.IConditionSeq`

Bases: `pybind11_object`

__add__ (*self*: rti.connextdds.IConditionSeq, *arg0*: rti.connextdds.IConditionSeq) → `rti.connextdds.IConditionSeq`

__bool__ (*self*: rti.connextdds.IConditionSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connextdds.IConditionSeq, *x*: rti.connextdds.ICondition) → bool

Return true the container contains *x*

`__delitem__` (*args, **kwargs)

Overloaded function.

1. `__delitem__(self: rti.connexdds.IConditionSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connexdds.IConditionSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (self: rti.connexdds.IConditionSeq, arg0: rti.connexdds.IConditionSeq) → bool

`__getitem__` (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connexdds.IConditionSeq, s: slice) -> rti.connexdds.IConditionSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.IConditionSeq, arg0: int) -> rti.connexdds.ICondition`

`__hash__` = None

`__iadd__` (self: rti.connexdds.IConditionSeq, arg0: rti.connexdds.IConditionSeq) → *rti.connexdds.IConditionSeq*

`__imul__` (self: rti.connexdds.IConditionSeq, arg0: int) → *rti.connexdds.IConditionSeq*

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.IConditionSeq) -> None`
2. `__init__(self: rti.connexdds.IConditionSeq, arg0: rti.connexdds.IConditionSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.IConditionSeq, arg0: Iterable) -> None`

`__iter__` (self: rti.connexdds.IConditionSeq) → Iterator

`__len__` (self: rti.connexdds.IConditionSeq) → int

`__module__` = 'rti.connexdds'

`__mul__` (self: rti.connexdds.IConditionSeq, arg0: int) → *rti.connexdds.IConditionSeq*

`__ne__` (self: rti.connexdds.IConditionSeq, arg0: rti.connexdds.IConditionSeq) → bool

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` = <capsule object NULL>

`__repr__` (self: rti.connexdds.IConditionSeq) → str

Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.IConditionSeq, *arg0*: int) → rti.connexdds.IConditionSeq

__setitem__ (*args, **kwargs)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.IConditionSeq, *arg0*: int, *arg1*: rti.connexdds.ICondition) → None
2. **__setitem__**(*self*: rti.connexdds.IConditionSeq, *arg0*: slice, *arg1*: rti.connexdds.IConditionSeq) → None

Assign list elements using a slice object

append (*self*: rti.connexdds.IConditionSeq, *x*: rti.connexdds.ICondition) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.IConditionSeq) → None

Clear the contents

count (*self*: rti.connexdds.IConditionSeq, *x*: rti.connexdds.ICondition) → int

Return the number of times *x* appears in the list

extend (*args, **kwargs)

Overloaded function.

1. **extend**(*self*: rti.connexdds.IConditionSeq, *L*: rti.connexdds.IConditionSeq) → None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.IConditionSeq, *L*: Iterable) → None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.IConditionSeq, *i*: int, *x*: rti.connexdds.ICondition) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. **pop**(*self*: rti.connexdds.IConditionSeq) → rti.connexdds.ICondition

Remove and return the last item

2. **pop**(*self*: rti.connexdds.IConditionSeq, *i*: int) → rti.connexdds.ICondition

Remove and return the item at index *i*

remove (*self*: rti.connexdds.IConditionSeq, *x*: rti.connexdds.ICondition) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.IDataReader

Bases: *IEntity*, *IAnyDataReader*

__init__ (*args, **kwargs)


```
__module__ = 'rti.connexdds'
```

```
class rti.connexdds.IEntity
```

```
Bases: pybind11_object
```

This is the abstract base class for all the DDS objects that support QoS policies, a listener and a status condition.

```
__eq__ (self: rti.connexdds.IEntity, arg0: rti.connexdds.IEntity) → bool
    Test for equality.
```

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.IEntity, arg0: rti.connexdds.IEntity) → bool
    Test for inequality.
```

```
close (self: rti.connexdds.IEntity) → None
    Forces the destruction of this entity.
```

```
property closed
    Returns a boolean indicating if this Entity is closed.
```

```
enable (self: rti.connexdds.IEntity) → None
    Enables this entity (if it was created disabled).
```

```
property enabled
    Returns a boolean indicating if this Entity is enabled.
```

```
property instance_handle
    The instance handle that represents this entity.
```

```
retain (self: rti.connexdds.IEntity) → None
    Disables the automatic destruction of this entity.
```

```
property status_changes
    The list of communication statuses that are triggered.
```

```
unretain (self: rti.connexdds.IEntity) → None
    Decrement the retention count.
```

```
property use_count
    Returns the internal use count value for this Entity.
```

```
class rti.connexdds.IEntitySeq
```

```
Bases: pybind11_object
```

```
__add__ (self: rti.connexdds.IEntitySeq, arg0: rti.connexdds.IEntitySeq) →
    rti.connexdds.IEntitySeq
```

__bool__ (*self*: rti.connexdds.IEntitySeq) → bool
 Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.IEntitySeq, *x*: rti.connexdds.IEntity) → bool
 Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.IEntitySeq, *arg0*: int) -> None
 Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connexdds.IEntitySeq, *arg0*: slice) -> None
 Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.IEntitySeq, *arg0*: rti.connexdds.IEntitySeq) → bool

__getitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.IEntitySeq, *s*: slice) -> rti.connexdds.IEntitySeq
 Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.IEntitySeq, *arg0*: int) -> rti.connexdds.IEntity

__hash__ = None

__iadd__ (*self*: rti.connexdds.IEntitySeq, *arg0*: rti.connexdds.IEntitySeq) → *rti.connexdds.IEntitySeq*

__imul__ (*self*: rti.connexdds.IEntitySeq, *arg0*: int) → *rti.connexdds.IEntitySeq*

__init__ (**args*, ***kwargs*)
 Overloaded function.

1. **__init__**(*self*: rti.connexdds.IEntitySeq) -> None
2. **__init__**(*self*: rti.connexdds.IEntitySeq, *arg0*: rti.connexdds.IEntitySeq) -> None
 Copy constructor
3. **__init__**(*self*: rti.connexdds.IEntitySeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.IEntitySeq) → Iterator

__len__ (*self*: rti.connexdds.IEntitySeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.IEntitySeq, *arg0*: int) → *rti.connexdds.IEntitySeq*

__ne__ (*self*: rti.connexdds.IEntitySeq, *arg0*: rti.connexdds.IEntitySeq) → bool

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` = <capsule object NULL>

`__repr__` (*self*: rti.connexdds.IEntitySeq) → str

Return the canonical string representation of this list.

`__rmul__` (*self*: rti.connexdds.IEntitySeq, *arg0*: int) → rti.connexdds.IEntitySeq

`__setitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__setitem__`(*self*: rti.connexdds.IEntitySeq, *arg0*: int, *arg1*: rti.connexdds.IEntity) -> None
2. `__setitem__`(*self*: rti.connexdds.IEntitySeq, *arg0*: slice, *arg1*: rti.connexdds.IEntitySeq) -> None

Assign list elements using a slice object

`append` (*self*: rti.connexdds.IEntitySeq, *x*: rti.connexdds.IEntity) → None

Add an item to the end of the list

`clear` (*self*: rti.connexdds.IEntitySeq) → None

Clear the contents

`count` (*self*: rti.connexdds.IEntitySeq, *x*: rti.connexdds.IEntity) → int

Return the number of times *x* appears in the list

`extend` (**args*, ***kwargs*)

Overloaded function.

1. `extend`(*self*: rti.connexdds.IEntitySeq, *L*: rti.connexdds.IEntitySeq) -> None

Extend the list by appending all the items in the given list

2. `extend`(*self*: rti.connexdds.IEntitySeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

`insert` (*self*: rti.connexdds.IEntitySeq, *i*: int, *x*: rti.connexdds.IEntity) → None

Insert an item at a given position.

`pop` (**args*, ***kwargs*)

Overloaded function.

1. `pop`(*self*: rti.connexdds.IEntitySeq) -> rti.connexdds.IEntity

Remove and return the last item

2. `pop`(*self*: rti.connexdds.IEntitySeq, *i*: int) -> rti.connexdds.IEntity

Remove and return the item at index *i*

`remove` (*self*: rti.connexdds.IEntitySeq, *x*: rti.connexdds.IEntity) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

```

class rti.connexdds.IReadCondition
    Bases: ICondition

    __enter__ (self: rti.connexdds.IReadCondition) → rti.connexdds.IReadCondition

    __eq__ (self: rti.connexdds.IReadCondition, arg0: rti.connexdds.IReadCondition) → bool
        Test for equality.

    __exit__ (self: rti.connexdds.IReadCondition, arg0: object, arg1: object, arg2: object) → None

    __hash__ = None

    __init__ (*args, **kwargs)

    __module__ = 'rti.connexdds'

    __ne__ (self: rti.connexdds.IReadCondition, arg0: rti.connexdds.IReadCondition) → bool
        Test for inequality.

    close (self: rti.connexdds.IReadCondition) → None
        Returns the DataReader associated to this condition.

    property closed
        Returns the DataReader associated to this condition.

    property data_reader
        Returns the DataReader associated to this condition.

    property state_filter
        Returns the DataState of this condition.

class rti.connexdds.ITopicDescription
    Bases: IEntity

    __init__ (*args, **kwargs)

    __module__ = 'rti.connexdds'

    property name
        The name of the entity conforming to the ITopicDescription interface.

    property participant
        The parent DomainParticipant.

    property type_name
        The name of the associated type.

class rti.connexdds.IgnoredEntityReplacementKind
    Bases: pybind11_object

```

```

class IgnoredEntityReplacementKind
    Bases: pybind11_object

    Members:

    NO_REPLACEMENT

    NOT_ALIVE_FIRST

    NOT_ALIVE_FIRST =
    <IgnoredEntityReplacementKind.NOT_ALIVE_FIRST: 1>

    NO_REPLACEMENT = <IgnoredEntityReplacementKind.NO_REPLACEMENT:
    0>

    __eq__(self: object, other: object) → bool

    __getstate__(self: object) → int

    __hash__(self: object) → int

    __index__(self:
        rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind)
        → int

    __init__(self:
        rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind,
        value: int) → None

    __int__(self:
        rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind) →
        int

    __members__ = {'NOT_ALIVE_FIRST':
    <IgnoredEntityReplacementKind.NOT_ALIVE_FIRST: 1>,
    'NO_REPLACEMENT': <IgnoredEntityReplacementKind.NO_REPLACEMENT:
    0>}

    __module__ = 'rti.connexdds'

    __ne__(self: object, other: object) → bool

    __repr__(self: object) → str

    __setstate__(self: rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplac-
        ementKind, state: int) →
        None

    __str__()
        name(self: handle) -> str

    property name

```

property value

NOT_ALIVE_FIRST = <IgnoredEntityReplacementKind.NOT_ALIVE_FIRST: 1>

NO_REPLACEMENT = <IgnoredEntityReplacementKind.NO_REPLACEMENT: 0>

__eq__ (*self*: rti.connexdds.IgnoredEntityReplacementKind, *arg0*: rti.connexdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.IgnoredEntityReplacementKind, *arg0*: rti.connexdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.IgnoredEntityReplacementKind, *arg0*: rti.connexdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.IgnoredEntityReplacementKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.IgnoredEntityReplacementKind, *arg0*: rti.connexdds.IgnoredEntityReplacementKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.IgnoredEntityReplacementKind) → *rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind*

Int conversion.

__le__ (*self*: rti.connexdds.IgnoredEntityReplacementKind, *arg0*: rti.connexdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.IgnoredEntityReplacementKind, *arg0*: rti.connexdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.IgnoredEntityReplacementKind, *arg0*: rti.connexdds.IgnoredEntityReplacementKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.IgnoredEntityReplacementKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

exception rti.connexdds.IllegalOperationError

Bases: *Exception*

__module__ = 'rti.connexdds'

exception rti.connexdds.ImmutablePolicyError

Bases: *Exception*

__module__ = 'rti.connexdds'

exception rti.connexdds.InconsistentPolicyError

Bases: *Exception*

__module__ = 'rti.connexdds'

class rti.connexdds.InconsistentTopicStatus

Bases: *pybind11_object*

__init__ (*args, **kwargs)

__module__ = 'rti.connexdds'

property total_count

Get the total count of pairs of DataReaders/DataWriters whose Topic names match but data types are inconsistent according to the current type consistency enforcement policy.

property total_count_change

The delta number of inconsistent pairs of DataReaders/DataWriters for the Topic that have been discovered since the last time this status was read.

class rti.connexdds.InstanceHandle

Bases: *pybind11_object*

__bool__ (self: rti.connexdds.InstanceHandle) → bool

__eq__ (self: rti.connexdds.InstanceHandle, arg0: rti.connexdds.InstanceHandle) → bool

Test for equality.

__hash__ (self: rti.connexdds.InstanceHandle) → int

__init__ (self: rti.connexdds.InstanceHandle) → None

Create a nil InstanceHandle

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.InstanceHandle, arg0: rti.connexdds.InstanceHandle) → bool

Test for inequality.

__nonzero__ (self: rti.connexdds.InstanceHandle) → bool

__str__ (*self*: rti.connexdds.InstanceHandle) → str

property is_nil

Nil status of InstanceStatus

static nil () → *rti.connexdds.InstanceHandle*

Create a nil InstanceHandle.

class rti.connexdds.InstanceHandleSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.InstanceHandleSeq, *arg0*: rti.connexdds.InstanceHandleSeq) → *rti.connexdds.InstanceHandleSeq*

__bool__ (*self*: rti.connexdds.InstanceHandleSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.InstanceHandleSeq, *x*: rti.connexdds.InstanceHandle) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.InstanceHandleSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.InstanceHandleSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.InstanceHandleSeq, *arg0*: rti.connexdds.InstanceHandleSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.InstanceHandleSeq, *s*: slice) -> rti.connexdds.InstanceHandleSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.InstanceHandleSeq, *arg0*: int) -> rti.connexdds.InstanceHandle

__hash__ = None

__iadd__ (*self*: rti.connexdds.InstanceHandleSeq, *arg0*: rti.connexdds.InstanceHandleSeq) → *rti.connexdds.InstanceHandleSeq*

__imul__ (*self*: rti.connexdds.InstanceHandleSeq, *arg0*: int) → *rti.connexdds.InstanceHandleSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.InstanceHandleSeq) -> None

2. `__init__(self: rti.connextdds.InstanceHandleSeq, arg0: rti.connextdds.InstanceHandleSeq)`
-> None

Copy constructor

3. `__init__(self: rti.connextdds.InstanceHandleSeq, arg0: Iterable)` -> None

`__iter__(self: rti.connextdds.InstanceHandleSeq)` → Iterator

`__len__(self: rti.connextdds.InstanceHandleSeq)` → int

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.InstanceHandleSeq, arg0: int)` → *rti.connextdds.InstanceHandleSeq*

`__ne__(self: rti.connextdds.InstanceHandleSeq, arg0: rti.connextdds.InstanceHandleSeq)` → bool

`__repr__(self: rti.connextdds.InstanceHandleSeq)` → str

Return the canonical string representation of this list.

`__rmul__(self: rti.connextdds.InstanceHandleSeq, arg0: int)` → *rti.connextdds.InstanceHandleSeq*

`__setitem__(*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connextdds.InstanceHandleSeq, arg0: int, arg1: rti.connextdds.InstanceHandle)` -> None
2. `__setitem__(self: rti.connextdds.InstanceHandleSeq, arg0: slice, arg1: rti.connextdds.InstanceHandleSeq)` -> None

Assign list elements using a slice object

`append(self: rti.connextdds.InstanceHandleSeq, x: rti.connextdds.InstanceHandle)` → None

Add an item to the end of the list

`clear(self: rti.connextdds.InstanceHandleSeq)` → None

Clear the contents

`count(self: rti.connextdds.InstanceHandleSeq, x: rti.connextdds.InstanceHandle)` → int

Return the number of times x appears in the list

`extend(*args, **kwargs)`

Overloaded function.

1. `extend(self: rti.connextdds.InstanceHandleSeq, L: rti.connextdds.InstanceHandleSeq)` -> None

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.InstanceHandleSeq, L: Iterable)` -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.InstanceHandleSeq, *i*: int, *x*: rti.connexdds.InstanceHandle) → None
 Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.InstanceHandleSeq) -> rti.connexdds.InstanceHandle

Remove and return the last item

2. pop(*self*: rti.connexdds.InstanceHandleSeq, *i*: int) -> rti.connexdds.InstanceHandle

Remove and return the item at index *i*

remove (*self*: rti.connexdds.InstanceHandleSeq, *x*: rti.connexdds.InstanceHandle) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.InstanceState

Bases: pybind11_object

ALIVE = <rti.connexdds.InstanceState object>

ANY = <rti.connexdds.InstanceState object>

NOT_ALIVE_DISPOSED = <rti.connexdds.InstanceState object>

NOT_ALIVE_MASK = <rti.connexdds.InstanceState object>

NOT_ALIVE_NO_WRITERS = <rti.connexdds.InstanceState object>

__and__ (*self*: rti.connexdds.InstanceState, *arg0*: rti.connexdds.InstanceState) →
 rti.connexdds.InstanceState

Bitwise logical AND of masks.

__bool__ (*self*: rti.connexdds.InstanceState) → rti.connexdds.InstanceState

Test if any bits are set.

__contains__ (*self*: rti.connexdds.InstanceState, *arg0*: rti.connexdds.InstanceState) → bool

__eq__ (*self*: rti.connexdds.InstanceState, *arg0*: rti.connexdds.InstanceState) → bool

Compare masks for equality.

__getitem__ (*self*: rti.connexdds.InstanceState, *arg0*: int) → bool

Get individual mask bit.

__hash__ = None

__iand__ (*self*: rti.connexdds.InstanceState, *arg0*: rti.connexdds.InstanceState) →
 rti.connexdds.InstanceState

Set mask to logical AND with another mask.

__ilshift__ (*self*: rti.connexdds.InstanceState, *arg0*: int) → rti.connexdds.InstanceState

Left shift bits in mask.

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.InstanceState) -> None

Create an InstanceState with no bits set.

2. **__init__**(self: rti.connexdds.InstanceState, value: int) -> None

Creates a mask from the bits in an integer.

__int__ (self: rti.connexdds.InstanceState) → int

Convert mask to int.

__ior__ (self: rti.connexdds.InstanceState, arg0: rti.connexdds.InstanceState) → *rti.connexdds.InstanceState*

Set mask to logical OR with another mask.

__irshift__ (self: rti.connexdds.InstanceState, arg0: int) → *rti.connexdds.InstanceState*

Right shift bits in mask.

__ixor__ (self: rti.connexdds.InstanceState, arg0: rti.connexdds.InstanceState) → *rti.connexdds.InstanceState*

Set mask to logical XOR with another mask.

__lshift__ (self: rti.connexdds.InstanceState, arg0: int) → *rti.connexdds.InstanceState*

Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.InstanceState, arg0: rti.connexdds.InstanceState) → bool

Compare masks for inequality.

__or__ (self: rti.connexdds.InstanceState, arg0: rti.connexdds.InstanceState) → *rti.connexdds.InstanceState*

Bitwise logical OR of masks.

__rshift__ (self: rti.connexdds.InstanceState, arg0: int) → *rti.connexdds.InstanceState*

Right shift bits in mask.

__setitem__ (self: rti.connexdds.InstanceState, arg0: int, arg1: bool) → None

Set individual mask bit

__str__ (self: rti.connexdds.InstanceState) → str

__xor__ (self: rti.connexdds.InstanceState, arg0: rti.connexdds.InstanceState) → *rti.connexdds.InstanceState*

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (*args, **kwargs)

Overloaded function.

1. flip(self: rti.connextdds.InstanceState) -> rti.connextdds.InstanceState

Flip all bits in the mask.

2. flip(self: rti.connextdds.InstanceState, pos: int) -> rti.connextdds.InstanceState

Flip the mask bit at the specified position.

reset (*args, **kwargs)

Overloaded function.

1. reset(self: rti.connextdds.InstanceState) -> rti.connextdds.InstanceState

Clear all bits in the mask.

2. reset(self: rti.connextdds.InstanceState, pos: int) -> rti.connextdds.InstanceState

Clear the mask bit at the specified position.

set (*args, **kwargs)

Overloaded function.

1. set(self: rti.connextdds.InstanceState) -> rti.connextdds.InstanceState

Set all bits in the mask.

2. set(self: rti.connextdds.InstanceState, pos: int, value: bool = True) -> rti.connextdds.InstanceState

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (self: rti.connextdds.InstanceState, pos: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (self: rti.connextdds.InstanceState) → bool

Test if all bits are set.

test_any (self: rti.connextdds.InstanceState) → bool

Test if any bits are set.

test_none (self: rti.connextdds.InstanceState) → bool

Test if none of the bits are set.

class rti.connextdds.InstanceStateConsistencyKind

Bases: pybind11_object

Members:

NONE : Instance states on the DataReader are not guaranteed to be correct after liveliness is regained after a disconnect

RECOVER_STATE : Instance states on the DataReader are guaranteed to be correct after liveliness is regained after a disconnect.

NONE = <InstanceStateConsistencyKind.NONE: 0>

RECOVER_STATE = <InstanceStateConsistencyKind.RECOVER_STATE: 1>

__eq__ (self: object, other: object) → bool

__getstate__ (self: object) → int

__hash__ (self: object) → int

__index__ (self: rti.connexdds.InstanceStateConsistencyKind) → int

__init__ (self: rti.connexdds.InstanceStateConsistencyKind, value: int) → None

__int__ (self: rti.connexdds.InstanceStateConsistencyKind) → int

__members__ = {'NONE': <InstanceStateConsistencyKind.NONE: 0>, 'RECOVER_STATE': <InstanceStateConsistencyKind.RECOVER_STATE: 1>}

__module__ = 'rti.connexdds'

__ne__ (self: object, other: object) → bool

__repr__ (self: object) → str

__setstate__ (self: rti.connexdds.InstanceStateConsistencyKind, state: int) → None

__str__ ()

name(self: handle) -> str

property name

property value

class rti.connexdds.Int16Seq

Bases: pybind11_object

__add__ (self: rti.connexdds.Int16Seq, arg0: rti.connexdds.Int16Seq) → rti.connexdds.Int16Seq

__bool__ (self: rti.connexdds.Int16Seq) → bool

Check whether the list is nonempty

__contains__ (self: rti.connexdds.Int16Seq, x: int) → bool

Return true the container contains x

__delitem__ (*args, **kwargs)

Overloaded function.

1. **__delitem__**(self: rti.connexdds.Int16Seq, arg0: int) -> None

Delete the list elements at index i

2. `__delitem__(self: rti.connexdds.Int16Seq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__(self: rti.connexdds.Int16Seq, arg0: rti.connexdds.Int16Seq) → bool`

`__getitem__(*args, **kwargs)`

Overloaded function.

1. `__getitem__(self: rti.connexdds.Int16Seq, s: slice) -> rti.connexdds.Int16Seq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.Int16Seq, arg0: int) -> int`

`__getstate__(self: rti.connexdds.Int16Seq) → bytes`

`__hash__ = None`

`__iadd__(self: rti.connexdds.Int16Seq, arg0: rti.connexdds.Int16Seq) → rti.connexdds.Int16Seq`

`__imul__(self: rti.connexdds.Int16Seq, arg0: int) → rti.connexdds.Int16Seq`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connexdds.Int16Seq, arg0: buffer) -> None`

2. `__init__(self: rti.connexdds.Int16Seq) -> None`

3. `__init__(self: rti.connexdds.Int16Seq, arg0: rti.connexdds.Int16Seq) -> None`

Copy constructor

4. `__init__(self: rti.connexdds.Int16Seq, arg0: Iterable) -> None`

5. `__init__(self: rti.connexdds.Int16Seq, arg0: int) -> None`

`__iter__(self: rti.connexdds.Int16Seq) → Iterator`

`__len__(self: rti.connexdds.Int16Seq) → int`

`__module__ = 'rti.connexdds'`

`__mul__(self: rti.connexdds.Int16Seq, arg0: int) → rti.connexdds.Int16Seq`

`__ne__(self: rti.connexdds.Int16Seq, arg0: rti.connexdds.Int16Seq) → bool`

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>`

`__repr__(self: rti.connexdds.Int16Seq) → str`

Return the canonical string representation of this list.

`__rmul__(self: rti.connexdds.Int16Seq, arg0: int) → rti.connexdds.Int16Seq`

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.Int16Seq, arg0: int, arg1: int) -> None`
2. `__setitem__(self: rti.connextdds.Int16Seq, arg0: slice, arg1: rti.connextdds.Int16Seq) -> None`

Assign list elements using a slice object

__setstate__ (self: rti.connextdds.Int16Seq, arg0: bytes) → None

append (self: rti.connextdds.Int16Seq, x: int) → None

Add an item to the end of the list

clear (self: rti.connextdds.Int16Seq) → None

Clear the contents

count (self: rti.connextdds.Int16Seq, x: int) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connextdds.Int16Seq, L: rti.connextdds.Int16Seq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.Int16Seq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.Int16Seq, i: int, x: int) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.Int16Seq) -> int`

Remove and return the last item

2. `pop(self: rti.connextdds.Int16Seq, i: int) -> int`

Remove and return the item at index i

remove (self: rti.connextdds.Int16Seq, x: int) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

resize (self: rti.connextdds.Int16Seq, arg0: int) → None

resizes the vector to the given size

class `rti.connextdds.Int16Type`

Bases: *DynamicType*

`__eq__` (*self*: rti.connexdds.Int16Type, *arg0*: rti.connexdds.Int16Type) → bool
Test for equality.

`__hash__` = None

`__init__` (*self*: rti.connexdds.Int16Type) → None
Get the singleton for Int16Type

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.Int16Type, *arg0*: rti.connexdds.Int16Type) → bool
Test for inequality.

class rti.connexdds.Int32Seq

Bases: pybind11_object

`__add__` (*self*: rti.connexdds.Int32Seq, *arg0*: rti.connexdds.Int32Seq) → rti.connexdds.Int32Seq

`__bool__` (*self*: rti.connexdds.Int32Seq) → bool
Check whether the list is nonempty

`__contains__` (*self*: rti.connexdds.Int32Seq, *x*: int) → bool
Return true the container contains x

`__delitem__` (*args, **kwargs)
Overloaded function.

1. `__delitem__`(*self*: rti.connexdds.Int32Seq, *arg0*: int) -> None

Delete the list elements at index *i*

2. `__delitem__`(*self*: rti.connexdds.Int32Seq, *arg0*: slice) -> None

Delete list elements using a slice object

`__eq__` (*self*: rti.connexdds.Int32Seq, *arg0*: rti.connexdds.Int32Seq) → bool

`__getitem__` (*args, **kwargs)
Overloaded function.

1. `__getitem__`(*self*: rti.connexdds.Int32Seq, *s*: slice) -> rti.connexdds.Int32Seq

Retrieve list elements using a slice object

2. `__getitem__`(*self*: rti.connexdds.Int32Seq, *arg0*: int) -> int

`__getstate__` (*self*: rti.connexdds.Int32Seq) → bytes

`__hash__` = None

`__iadd__` (*self*: rti.connexdds.Int32Seq, *arg0*: rti.connexdds.Int32Seq) → rti.connexdds.Int32Seq

`__imul__` (*self*: rti.connexdds.Int32Seq, *arg0*: int) → rti.connexdds.Int32Seq

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.Int32Seq, arg0: buffer) -> None
2. **__init__**(self: rti.connexdds.Int32Seq) -> None
3. **__init__**(self: rti.connexdds.Int32Seq, arg0: rti.connexdds.Int32Seq) -> None

Copy constructor

4. **__init__**(self: rti.connexdds.Int32Seq, arg0: Iterable) -> None
5. **__init__**(self: rti.connexdds.Int32Seq, arg0: int) -> None

__iter__ (self: rti.connexdds.Int32Seq) → Iterator

__len__ (self: rti.connexdds.Int32Seq) → int

__module__ = 'rti.connexdds'

__mul__ (self: rti.connexdds.Int32Seq, arg0: int) → rti.connexdds.Int32Seq

__ne__ (self: rti.connexdds.Int32Seq, arg0: rti.connexdds.Int32Seq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (self: rti.connexdds.Int32Seq) → str

Return the canonical string representation of this list.

__rmul__ (self: rti.connexdds.Int32Seq, arg0: int) → rti.connexdds.Int32Seq

__setitem__ (*args, **kwargs)

Overloaded function.

1. **__setitem__**(self: rti.connexdds.Int32Seq, arg0: int, arg1: int) -> None
2. **__setitem__**(self: rti.connexdds.Int32Seq, arg0: slice, arg1: rti.connexdds.Int32Seq) -> None

Assign list elements using a slice object

__setstate__ (self: rti.connexdds.Int32Seq, arg0: bytes) → None

append (self: rti.connexdds.Int32Seq, x: int) → None

Add an item to the end of the list

clear (self: rti.connexdds.Int32Seq) → None

Clear the contents

count (self: rti.connexdds.Int32Seq, x: int) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. extend(self: rti.connextdds.Int32Seq, L: rti.connextdds.Int32Seq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.Int32Seq, L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.Int32Seq, i: int, x: int) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connextdds.Int32Seq) -> int

Remove and return the last item

2. pop(self: rti.connextdds.Int32Seq, i: int) -> int

Remove and return the item at index i

remove (self: rti.connextdds.Int32Seq, x: int) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

resize (self: rti.connextdds.Int32Seq, arg0: int) → None

resizes the vector to the given size

class rti.connextdds.Int32Type

Bases: *DynamicType*

__eq__ (self: rti.connextdds.Int32Type, arg0: rti.connextdds.Int32Type) → bool

Test for equality.

__hash__ = None

__init__ (self: rti.connextdds.Int32Type) → None

Get the singleton for Int32Type

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.Int32Type, arg0: rti.connextdds.Int32Type) → bool

Test for inequality.

class rti.connextdds.Int32Vector

Bases: *pybind11_object*

A DDS standard container with functionality similar to a C++ vector.

__eq__ (self: rti.connextdds.Int32Vector, arg0: rti.connextdds.Int32Vector) → bool

Compare Int32Vectors for equality.

__getitem__ (*self*: rti.connexdds.Int32Vector, *arg0*: int) → int

Get the value at the specified index.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.Int32Vector, *buffer*: buffer) -> None

Create a Int32Vector from another Python buffer.

2. **__init__**(*self*: rti.connexdds.Int32Vector) -> None

Create an empty Int32Vector

3. **__init__**(*self*: rti.connexdds.Int32Vector, *size*: int) -> None

Create a Int32Vector with a preallocated size.

4. **__init__**(*self*: rti.connexdds.Int32Vector, *vector*: rti.connexdds.Int32Vector) -> None

Create a copy from another Int32Vector.

5. **__init__**(*self*: rti.connexdds.Int32Vector, *arg0*: Iterable) -> None

6. **__init__**(*self*: rti.connexdds.Int32Vector, *list*: rti.connexdds.Int32Seq) -> None

Create a Int32Vector from a list of values.

__iter__ (*self*: rti.connexdds.Int32Vector) → Iterator

Iterate over the contents of the vector.

__len__ (*self*: rti.connexdds.Int32Vector) → int

Get the length of the Int32Vector.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.Int32Vector, *arg0*: rti.connexdds.Int32Vector) → bool

Compare Int32Vectors for inequality.

__setitem__ (*self*: rti.connexdds.Int32Vector, *arg0*: int, *arg1*: int) → None

Set the value at the specified index.

clear (*self*: rti.connexdds.Int32Vector) → None

Resize Int32Vector to 0.

resize (*self*: rti.connexdds.Int32Vector, *size*: int) → None

Resize Int32Vector.

class rti.connexdds.Int64Seq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.Int64Seq, *arg0*: rti.connexdds.Int64Seq) → rti.connexdds.Int64Seq

__bool__ (*self*: rti.connexdds.Int64Seq) → bool
 Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.Int64Seq, *x*: int) → bool
 Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.Int64Seq, *arg0*: int) -> None
 Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connexdds.Int64Seq, *arg0*: slice) -> None
 Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.Int64Seq, *arg0*: rti.connexdds.Int64Seq) → bool

__getitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.Int64Seq, *s*: slice) -> rti.connexdds.Int64Seq
 Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.Int64Seq, *arg0*: int) -> int

__getstate__ (*self*: rti.connexdds.Int64Seq) → bytes

__hash__ = None

__iadd__ (*self*: rti.connexdds.Int64Seq, *arg0*: rti.connexdds.Int64Seq) → *rti.connexdds.Int64Seq*

__imul__ (*self*: rti.connexdds.Int64Seq, *arg0*: int) → *rti.connexdds.Int64Seq*

__init__ (**args*, ***kwargs*)
 Overloaded function.

1. **__init__**(*self*: rti.connexdds.Int64Seq, *arg0*: buffer) -> None
2. **__init__**(*self*: rti.connexdds.Int64Seq) -> None
3. **__init__**(*self*: rti.connexdds.Int64Seq, *arg0*: rti.connexdds.Int64Seq) -> None
 Copy constructor
4. **__init__**(*self*: rti.connexdds.Int64Seq, *arg0*: Iterable) -> None
5. **__init__**(*self*: rti.connexdds.Int64Seq, *arg0*: int) -> None

__iter__ (*self*: rti.connexdds.Int64Seq) → Iterator

__len__ (*self*: rti.connexdds.Int64Seq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.Int64Seq, *arg0*: int) → rti.connexdds.Int64Seq

__ne__ (*self*: rti.connexdds.Int64Seq, *arg0*: rti.connexdds.Int64Seq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (*self*: rti.connexdds.Int64Seq) → str

Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.Int64Seq, *arg0*: int) → rti.connexdds.Int64Seq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.Int64Seq, *arg0*: int, *arg1*: int) -> None

2. **__setitem__**(*self*: rti.connexdds.Int64Seq, *arg0*: slice, *arg1*: rti.connexdds.Int64Seq) -> None

Assign list elements using a slice object

__setstate__ (*self*: rti.connexdds.Int64Seq, *arg0*: bytes) → None

append (*self*: rti.connexdds.Int64Seq, *x*: int) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.Int64Seq) → None

Clear the contents

count (*self*: rti.connexdds.Int64Seq, *x*: int) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.Int64Seq, *L*: rti.connexdds.Int64Seq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.Int64Seq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.Int64Seq, *i*: int, *x*: int) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.Int64Seq) -> int

Remove and return the last item

2. **pop**(*self*: rti.connexdds.Int64Seq, *i*: int) -> int

Remove and return the item at index *i*

remove (*self*: rti.connexdds.Int64Seq, *x*: int) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

resize (*self*: rti.connexdds.Int64Seq, *arg0*: int) → None

resizes the vector to the given size

class rti.connexdds.Int64Type

Bases: *DynamicType*

__eq__ (*self*: rti.connexdds.Int64Type, *arg0*: rti.connexdds.Int64Type) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.Int64Type) → None

Get the singleton for Int64Type

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.Int64Type, *arg0*: rti.connexdds.Int64Type) → bool

Test for inequality.

class rti.connexdds.Int8Seq

Bases: *pybind11_object*

__add__ (*self*: rti.connexdds.Int8Seq, *arg0*: rti.connexdds.Int8Seq) → *rti.connexdds.Int8Seq*

__bool__ (*self*: rti.connexdds.Int8Seq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.Int8Seq, *x*: int) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.Int8Seq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.Int8Seq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.Int8Seq, *arg0*: rti.connexdds.Int8Seq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.Int8Seq, *s*: slice) -> rti.connexdds.Int8Seq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.Int8Seq, *arg0*: int) -> int

__getstate__ (*self*: rti.connexdds.Int8Seq) → bytes

__hash__ = None

__iadd__ (*self*: rti.connexdds.Int8Seq, *arg0*: rti.connexdds.Int8Seq) → *rti.connexdds.Int8Seq*

__imul__ (*self*: rti.connexdds.Int8Seq, *arg0*: int) → *rti.connexdds.Int8Seq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.Int8Seq, *arg0*: buffer) -> None
2. **__init__**(*self*: rti.connexdds.Int8Seq) -> None
3. **__init__**(*self*: rti.connexdds.Int8Seq, *arg0*: rti.connexdds.Int8Seq) -> None

Copy constructor

4. **__init__**(*self*: rti.connexdds.Int8Seq, *arg0*: Iterable) -> None
5. **__init__**(*self*: rti.connexdds.Int8Seq, *arg0*: int) -> None

__iter__ (*self*: rti.connexdds.Int8Seq) → Iterator

__len__ (*self*: rti.connexdds.Int8Seq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.Int8Seq, *arg0*: int) → *rti.connexdds.Int8Seq*

__ne__ (*self*: rti.connexdds.Int8Seq, *arg0*: rti.connexdds.Int8Seq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (*self*: rti.connexdds.Int8Seq) → str

Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.Int8Seq, *arg0*: int) → *rti.connexdds.Int8Seq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.Int8Seq, *arg0*: int, *arg1*: int) -> None
2. **__setitem__**(*self*: rti.connexdds.Int8Seq, *arg0*: slice, *arg1*: rti.connexdds.Int8Seq) -> None

Assign list elements using a slice object

__setstate__ (*self*: rti.connexdds.Int8Seq, *arg0*: bytes) → None

append (*self*: rti.connexdds.Int8Seq, *x*: int) → None

Add an item to the end of the list

clear (*self*: rti.connextdds.Int8Seq) → None

Clear the contents

count (*self*: rti.connextdds.Int8Seq, *x*: int) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connextdds.Int8Seq, *L*: rti.connextdds.Int8Seq) -> None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connextdds.Int8Seq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.Int8Seq, *i*: int, *x*: int) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connextdds.Int8Seq) -> int

Remove and return the last item

2. pop(*self*: rti.connextdds.Int8Seq, *i*: int) -> int

Remove and return the item at index *i*

remove (*self*: rti.connextdds.Int8Seq, *x*: int) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

resize (*self*: rti.connextdds.Int8Seq, *arg0*: int) → None

resizes the vector to the given size

class rti.connextdds.Int8Type

Bases: *DynamicType*

__eq__ (*self*: rti.connextdds.Int8Type, *arg0*: rti.connextdds.Int8Type) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.Int8Type) → None

Get the singleton for Int8Type

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.Int8Type, *arg0*: rti.connextdds.Int8Type) → bool

Test for inequality.

exception rti.connextdds.InvalidArgumentError

Bases: *Exception*


```

__module__ = 'rti.connextdds'

exception rti.connextdds.InvalidDowncastError
    Bases: Exception

__module__ = 'rti.connextdds'

class rti.connextdds.InvalidLocalIdentityAdvanceNoticeStatus
    Bases: pybind11_object

    __init__(*args, **kwargs)

__module__ = 'rti.connextdds'

property expiration_time
    Get the expiration time of the status

class rti.connextdds.LatencyBudget
    Bases: pybind11_object

    __eq__(self: rti.connextdds.LatencyBudget, arg0: rti.connextdds.LatencyBudget) → bool
        Test for equality.

    __hash__ = None

    __init__(*args, **kwargs)
        Overloaded function.

        1. __init__(self: rti.connextdds.LatencyBudget) -> None
            Creates a latency budget with zero duration.

        2. __init__(self: rti.connextdds.LatencyBudget, duration: rti.connextdds.Duration) -> None
            Creates an instance with the specified duration.

__module__ = 'rti.connextdds'

__ne__(self: rti.connextdds.LatencyBudget, arg0: rti.connextdds.LatencyBudget) → bool
    Test for inequality.

property duration
    The duration of the maximum acceptable delay.

    This property's getter returns a deep copy.

class rti.connextdds.Lifespan
    Bases: pybind11_object

    __eq__(self: rti.connextdds.Lifespan, arg0: rti.connextdds.Lifespan) → bool
        Test for equality.

    __hash__ = None

```

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.Lifespan) -> None

Creates the default policy with an infinite lifespan.

2. **__init__**(self: rti.connexdds.Lifespan, duration: rti.connexdds.Duration) -> None

Creates a policy with the specified lifespan duration.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.Lifespan, arg0: rti.connexdds.Lifespan) → bool

Test for inequality.

property duration

Sets the maximum duration for which the data is valid.

This property's getter returns a deep copy.

class rti.connexdds.Liveliness

Bases: pybind11_object

__eq__ (self: rti.connexdds.Liveliness, arg0: rti.connexdds.Liveliness) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.Liveliness) -> None

Creates an automatic liveliness policy with infinite lease duration.

2. **__init__**(self: rti.connexdds.Liveliness, kind: rti.connexdds.LivelinessKind, duration: rti.connexdds.Duration) -> None

Creates an instance with the specified liveliness kind and lease duration.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.Liveliness, arg0: rti.connexdds.Liveliness) → bool

Test for inequality.

property assertions_per_lease_duration

The number of assertions to send during the lease duration.

automatic = <rti.connexdds.Liveliness object>

property kind

The liveliness kind.

property lease_duration

The duration within which a Entity must be asserted or else it is considered not alive.

This property's getter returns a deep copy.

static manual_by_participant (*lease*: rti.connextdds.Duration = *Duration.infinite*) → *rti.connextdds.Liveliness*

Creates a Liveliness instance with LivelinessKind.MANUAL_BY_PARTICIPANT.

static manual_by_topic (*lease*: rti.connextdds.Duration = *Duration.infinite*) → *rti.connextdds.Liveliness*

Creates a Liveliness instance with LivelinessKind.MANUAL_BY_TOPIC.

class rti.connextdds.LivelinessChangedStatus

Bases: pybind11_object

__init__ (*args, **kwargs)

__module__ = 'rti.connextdds'

property alive_count

The number of currently alive DataWriters that write to the Topic of the DataReader.

property alive_count_change

The delta in the alive count since the last time the listener fired or the status was read.

property last_publication_handle

The instance handle of the DataWriter with the most recent change in liveliness.

property not_alive_count

The number of currently NOT_ALIVE DataWriters that write to the Topic of the DataReader.

property not_alive_count_change

The delta in the NOT_ALIVE count since the last time the listener fired or the status was read.

class rti.connextdds.LivelinessKind

Bases: pybind11_object

AUTOMATIC = <LivelinessKind.AUTOMATIC: 0>

class LivelinessKind

Bases: pybind11_object

Members:

AUTOMATIC : [default] The infrastructure will automatically signal liveliness for the DataWriter (s) at least as often as required by the DataWriter (S) lease_duration.

A DataWriter with this setting does not need to take any specific action in order to be considered 'alive.' The DataWriter is only 'not alive' when the participant to which it belongs terminates (gracefully or not), or when there is a network problem that prevents the current participant from contacting that remote participant.

MANUAL_BY_PARTICIPANT : RTI Connex will assume that as long as at least one DataWriter belonging to the DomainParticipant (or the DomainParticipant itself) has asserted its liveliness, then the other DataWriters belonging to that same DomainParticipant are also alive.

The user application takes responsibility to signal liveliness to RTI Connex either by calling DomainParticipant.assert_liveliness, or by calling DataWriter.assert_liveliness, or DataWriter.write() on any DataWriter belonging to the DomainParticipant.

MANUAL_BY_TOPIC : RTI Connex will only assume liveliness of the DataWriter if the application has asserted liveliness of that DataWriter itself.

The user application takes responsibility to signal liveliness to RTI Connex using the DataWriter.assert_liveliness method, or by writing some data.

AUTOMATIC = <LivelinessKind.AUTOMATIC: 0>

MANUAL_BY_PARTICIPANT = <LivelinessKind.MANUAL_BY_PARTICIPANT: 1>

MANUAL_BY_TOPIC = <LivelinessKind.MANUAL_BY_TOPIC: 2>

__eq__ (self: object, other: object) → bool

__getstate__ (self: object) → int

__hash__ (self: object) → int

__index__ (self: rti.connexdds.LivelinessKind.LivelinessKind) → int

__init__ (self: rti.connexdds.LivelinessKind.LivelinessKind, value: int) → None

__int__ (self: rti.connexdds.LivelinessKind.LivelinessKind) → int

__members__ = {'AUTOMATIC': <LivelinessKind.AUTOMATIC: 0>, 'MANUAL_BY_PARTICIPANT': <LivelinessKind.MANUAL_BY_PARTICIPANT: 1>, 'MANUAL_BY_TOPIC': <LivelinessKind.MANUAL_BY_TOPIC: 2>}

__module__ = 'rti.connexdds'

__ne__ (self: object, other: object) → bool

__repr__ (self: object) → str

__setstate__ (self: rti.connexdds.LivelinessKind.LivelinessKind, state: int) → None

__str__ ()

name(self: handle) -> str

property name

property value

MANUAL_BY_PARTICIPANT = <LivelinessKind.MANUAL_BY_PARTICIPANT: 1>

MANUAL_BY_TOPIC = <LivelinessKind.MANUAL_BY_TOPIC: 2>

__eq__ (*self*: rti.connexdds.LivelinessKind, *arg0*: rti.connexdds.LivelinessKind) → bool
Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.LivelinessKind, *arg0*: rti.connexdds.LivelinessKind) → bool
Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.LivelinessKind, *arg0*: rti.connexdds.LivelinessKind) → bool
Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.LivelinessKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.LivelinessKind, *arg0*: rti.connexdds.LivelinessKind.LivelinessKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.LivelinessKind) → *rti.connexdds.LivelinessKind.LivelinessKind*
Int conversion.

__le__ (*self*: rti.connexdds.LivelinessKind, *arg0*: rti.connexdds.LivelinessKind) → bool
Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.LivelinessKind, *arg0*: rti.connexdds.LivelinessKind) → bool
Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.LivelinessKind, *arg0*: rti.connexdds.LivelinessKind) → bool
Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.LivelinessKind) → str
String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.LivelinessLostStatus

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

property total_count

Total count of times that a previously alive DataWriter became not alive due to a failure to assert its liveness signal within the agreed lease duration.

property total_count_change

The delta number of times liveness has been lost since the last time the listener callback fired or this status was checked.

class `rti.connexdds.LoanedDynamicData`

Bases: `pybind11_object`

__enter__ (*self*: `rti.connexdds.LoanedDynamicData`) → `rti.connexdds.LoanedDynamicData`

Enter a context for the loaned field, loan returned on context exit.

__exit__ (*self*: `rti.connexdds.LoanedDynamicData`, *arg0*: *object*, *arg1*: *object*, *arg2*: *object*) → `None`

Exit the context for the loaned field, returning the resources.

__init__ (**args*, ***kwargs*)

__module__ = `'rti.connexdds'`

property data

Obtains the DynamicData object representing a member of a DynamicData object.

return_loan (*self*: `rti.connexdds.LoanedDynamicData`) → `None`

Explicitly return a dynamic data loan.

class `rti.connexdds.Locator`

Bases: `pybind11_object`

Type used to represent the addressing information needed to send a message to an RTPS Endpoint using one of the supported transports.

__eq__ (*self*: `rti.connexdds.Locator`, *arg0*: `rti.connexdds.Locator`) → `bool`

Compare Locators for equality.

__hash__ = `None`

__init__ (*self*: `rti.connexdds.Locator`, *kind*: `rti.connexdds.LocatorKind.LocatorKind`, *port*: *int*, *address*: `rti.connexdds.Uint8Seq`) → `None`

Construct a Locator with the provided kind, port, and address.

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.Locator`, *arg0*: `rti.connexdds.Locator`) → `bool`

Compare Locators for inequality.

property address

Get/set the address for this Locator.

invalid = `<rti.connexdds.Locator object>`

property kind

Get/set the kind for this Locator.

property port

Get/set the port for this Locator.

class `rti.connexdds.LocatorFilter`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.LocatorFilter`, *arg0*: `rti.connexdds.LocatorFilter`) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: `rti.connexdds.LocatorFilter`) -> None

Creates the default policy.

2. **__init__**(*self*: `rti.connexdds.LocatorFilter`, *locator_filters*: `rti.connexdds.LocatorFilterElementSeq`, *filter_name*: `str` = `Filter.stringmatch_filter_name`) -> None

Creates an instance with a sequence of filters and a filter name.

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.LocatorFilter`, *arg0*: `rti.connexdds.LocatorFilter`) → bool

Test for inequality.

property filter_name

The filter name.

property locator_filters

The locator filters.

This property's getter returns a deep copy.

class `rti.connexdds.LocatorFilterElement`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.LocatorFilterElement`, *arg0*: `rti.connexdds.LocatorFilterElement`) → bool

Test for equality.

__hash__ = None

__init__ (*self*: `rti.connexdds.LocatorFilterElement`, *arg0*: `str`, *arg1*: `rti.connexdds.LocatorSeq`) → None

Creates an instance with the provided `filter_expression` and `locators`.

__module__ = `'rti.connexdds'`

__ne__ (*self*: rti.connexdds.LocatorFilterElement, *arg0*: rti.connexdds.LocatorFilterElement) → bool

Test for inequality.

property filter_expression

The filter expression.

property locators

The locators associated with this filter.

This property's getter returns a deep copy.

class rti.connexdds.LocatorFilterElementSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: rti.connexdds.LocatorFilterElementSeq) → rti.connexdds.LocatorFilterElementSeq

__bool__ (*self*: rti.connexdds.LocatorFilterElementSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.LocatorFilterElementSeq, *x*: rti.connexdds.LocatorFilterElement) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: rti.connexdds.LocatorFilterElementSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.LocatorFilterElementSeq, *s*: slice) -> rti.connexdds.LocatorFilterElementSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: int) -> rti.connexdds.LocatorFilterElement

__hash__ = None

__iadd__ (*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: rti.connexdds.LocatorFilterElementSeq) → rti.connexdds.LocatorFilterElementSeq

__imul__ (*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: int) →
rti.connexdds.LocatorFilterElementSeq

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.LocatorFilterElementSeq) -> None
2. **__init__**(*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: rti.connexdds.LocatorFilterElementSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.LocatorFilterElementSeq) → Iterator

__len__ (*self*: rti.connexdds.LocatorFilterElementSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: int) →
rti.connexdds.LocatorFilterElementSeq

__ne__ (*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*:
rti.connexdds.LocatorFilterElementSeq) → bool

__rmul__ (*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: int) →
rti.connexdds.LocatorFilterElementSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: int, *arg1*: rti.connexdds.LocatorFilterElement) -> None
2. **__setitem__**(*self*: rti.connexdds.LocatorFilterElementSeq, *arg0*: slice, *arg1*: rti.connexdds.LocatorFilterElementSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.LocatorFilterElementSeq, *x*: rti.connexdds.LocatorFilterElement) →
None

Add an item to the end of the list

clear (*self*: rti.connexdds.LocatorFilterElementSeq) → None

Clear the contents

count (*self*: rti.connexdds.LocatorFilterElementSeq, *x*: rti.connexdds.LocatorFilterElement) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. `extend(self: rti.connextdds.LocatorFilterElementSeq, L: rti.connextdds.LocatorFilterElementSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.LocatorFilterElementSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.LocatorFilterElementSeq, *i*: int, *x*: rti.connextdds.LocatorFilterElement) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.LocatorFilterElementSeq) -> rti.connextdds.LocatorFilterElement`

Remove and return the last item

2. `pop(self: rti.connextdds.LocatorFilterElementSeq, i: int) -> rti.connextdds.LocatorFilterElement`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.LocatorFilterElementSeq, *x*: rti.connextdds.LocatorFilterElement) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connextdds.LocatorKind

Bases: pybind11_object

ANY = <LocatorKind.ANY: 0>

INTRA = <LocatorKind.INTRA: 3>

INVALID = <LocatorKind.INVALID: -1>

class LocatorKind

Bases: pybind11_object

Members:

INVALID : An invalid locator

ANY : A special value for any kind of locator.

UDPv4 : A locator for a UDPv4 address.

SHMEM : A locator for an address accessed via shared memory.

SHMEM_510 : A locator for an address accessed via shared memory with backwards compatibility for Connex 5.1.0 or earlier.

INTRA : A locator for the Connex INTRA transport.

UDPv6 : A locator for a UDPv6 address.

UDpv6_510 : A locator for a UDPv6 address with backwards compatibility for Connex 5.1.0 or earlier.

TCPV4_LAN : A locator for an address that communicates using TCP on a LAN.

TCPV4_WAN : A locator for an address that communicates using TCP over a WAN.

TLSV4_LAN : A locator for an address that communicates using TLS over a LAN.

TLSV4_WAN : A locator for an address that communicates using TLS over a WAN.

RESERVED : Reserved locator kind.

ANY = <LocatorKind.ANY: 0>

INTRA = <LocatorKind.INTRA: 3>

INVALID = <LocatorKind.INVALID: -1>

RESERVED = <LocatorKind.RESERVED: 1000>

SHMEM = <LocatorKind.SHMEM: 16777216>

SHMEM_510 = <LocatorKind.SHMEM_510: 2>

TCPV4_LAN = <LocatorKind.TCPV4_LAN: 8>

TCPV4_WAN = <LocatorKind.TCPV4_WAN: 9>

TLSV4_LAN = <LocatorKind.TLSV4_LAN: 10>

TLSV4_WAN = <LocatorKind.TLSV4_WAN: 11>

UDpv4 = <LocatorKind.UDpv4: 1>

UDpv6 = <LocatorKind.SHMEM_510: 2>

UDpv6_510 = <LocatorKind.UDpv6_510: 5>

__eq__ (*self: object, other: object*) → bool

__getstate__ (*self: object*) → int

__hash__ (*self: object*) → int

__index__ (*self: rti.connexdds.LocatorKind.LocatorKind*) → int

__init__ (*self: rti.connexdds.LocatorKind.LocatorKind, value: int*) → None

__int__ (*self: rti.connexdds.LocatorKind.LocatorKind*) → int

```

__members__ = {'ANY': <LocatorKind.ANY: 0>, 'INTRA':
<LocatorKind.INTRA: 3>, 'INVALID': <LocatorKind.INVALID: -1>,
'RESERVED': <LocatorKind.RESERVED: 1000>, 'SHMEM':
<LocatorKind.SHMEM: 16777216>, 'SHMEM_510':
<LocatorKind.SHMEM_510: 2>, 'TCPV4_LAN':
<LocatorKind.TCPV4_LAN: 8>, 'TCPV4_WAN':
<LocatorKind.TCPV4_WAN: 9>, 'TLSV4_LAN':
<LocatorKind.TLSV4_LAN: 10>, 'TLSV4_WAN':
<LocatorKind.TLSV4_WAN: 11>, 'UDPv4': <LocatorKind.UDPv4: 1>,
'UDPv6': <LocatorKind.SHMEM_510: 2>, 'UDPv6_510':
<LocatorKind.UDPv6_510: 5>}

```

```
__module__ = 'rti.connexdds'
```

```
__ne__(self: object, other: object) → bool
```

```
__repr__(self: object) → str
```

```
__setstate__(self: rti.connexdds.LocatorKind.LocatorKind, state: int) → None
```

```
__str__()
```

```
name(self: handle) -> str
```

property name

property value

```
RESERVED = <LocatorKind.RESERVED: 1000>
```

```
SHMEM = <LocatorKind.SHMEM: 16777216>
```

```
SHMEM_510 = <LocatorKind.SHMEM_510: 2>
```

```
TCPV4_LAN = <LocatorKind.TCPV4_LAN: 8>
```

```
TCPV4_WAN = <LocatorKind.TCPV4_WAN: 9>
```

```
TLSV4_LAN = <LocatorKind.TLSV4_LAN: 10>
```

```
TLSV4_WAN = <LocatorKind.TLSV4_WAN: 11>
```

```
UDPv4 = <LocatorKind.UDPv4: 1>
```

```
UDPv6 = <LocatorKind.SHMEM_510: 2>
```

```
UDPv6_510 = <LocatorKind.UDPv6_510: 5>
```

```
__eq__(self: rti.connexdds.LocatorKind, arg0: rti.connexdds.LocatorKind) → bool
```

Apply operator to underlying enumerated values.

```
__ge__(self: rti.connexdds.LocatorKind, arg0: rti.connexdds.LocatorKind) → bool
```

Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.LocatorKind, *arg0*: rti.connexdds.LocatorKind) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.LocatorKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.LocatorKind, *arg0*: rti.connexdds.LocatorKind.LocatorKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.LocatorKind) → *rti.connexdds.LocatorKind.LocatorKind*

Int conversion.

__le__ (*self*: rti.connexdds.LocatorKind, *arg0*: rti.connexdds.LocatorKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.LocatorKind, *arg0*: rti.connexdds.LocatorKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.LocatorKind, *arg0*: rti.connexdds.LocatorKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.LocatorKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.LocatorSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.LocatorSeq, *arg0*: rti.connexdds.LocatorSeq) → *rti.connexdds.LocatorSeq*

__bool__ (*self*: rti.connexdds.LocatorSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.LocatorSeq, *x*: rti.connexdds.Locator) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.LocatorSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. `__delitem__(self: rti.connexdds.LocatorSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__ (self: rti.connexdds.LocatorSeq, arg0: rti.connexdds.LocatorSeq) → bool`

`__getitem__ (*args, **kwargs)`

Overloaded function.

1. `__getitem__(self: rti.connexdds.LocatorSeq, s: slice) -> rti.connexdds.LocatorSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.LocatorSeq, arg0: int) -> rti.connexdds.Locator`

`__hash__ = None`

`__iadd__ (self: rti.connexdds.LocatorSeq, arg0: rti.connexdds.LocatorSeq) → rti.connexdds.LocatorSeq`

`__imul__ (self: rti.connexdds.LocatorSeq, arg0: int) → rti.connexdds.LocatorSeq`

`__init__ (*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connexdds.LocatorSeq) -> None`
2. `__init__(self: rti.connexdds.LocatorSeq, arg0: rti.connexdds.LocatorSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.LocatorSeq, arg0: Iterable) -> None`

`__iter__ (self: rti.connexdds.LocatorSeq) → Iterator`

`__len__ (self: rti.connexdds.LocatorSeq) → int`

`__module__ = 'rti.connexdds'`

`__mul__ (self: rti.connexdds.LocatorSeq, arg0: int) → rti.connexdds.LocatorSeq`

`__ne__ (self: rti.connexdds.LocatorSeq, arg0: rti.connexdds.LocatorSeq) → bool`

`__rmul__ (self: rti.connexdds.LocatorSeq, arg0: int) → rti.connexdds.LocatorSeq`

`__setitem__ (*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connexdds.LocatorSeq, arg0: int, arg1: rti.connexdds.Locator) -> None`
2. `__setitem__(self: rti.connexdds.LocatorSeq, arg0: slice, arg1: rti.connexdds.LocatorSeq) -> None`

Assign list elements using a slice object

append (*self*: rti.connexdds.LocatorSeq, *x*: rti.connexdds.Locator) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.LocatorSeq) → None

Clear the contents

count (*self*: rti.connexdds.LocatorSeq, *x*: rti.connexdds.Locator) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connexdds.LocatorSeq, *L*: rti.connexdds.LocatorSeq) -> None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connexdds.LocatorSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.LocatorSeq, *i*: int, *x*: rti.connexdds.Locator) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.LocatorSeq) -> rti.connexdds.Locator

Remove and return the last item

2. pop(*self*: rti.connexdds.LocatorSeq, *i*: int) -> rti.connexdds.Locator

Remove and return the item at index *i*

remove (*self*: rti.connexdds.LocatorSeq, *x*: rti.connexdds.Locator) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.LocatorVector

Bases: pybind11_object

A DDS standard container with functionality similar to a C++ vector.

__eq__ (*self*: rti.connexdds.LocatorVector, *arg0*: rti.connexdds.LocatorVector) → bool

Compare LocatorVectors for equality.

__getitem__ (*self*: rti.connexdds.LocatorVector, *arg0*: int) → rti.connexdds.Locator

Get the value at the specified index.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. __init__(*self*: rti.connexdds.LocatorVector) -> None

Create an empty LocatorVector

2. `__init__(self: rti.connextdds.LocatorVector, size: int) -> None`

Create a LocatorVector with a preallocated size.

3. `__init__(self: rti.connextdds.LocatorVector, vector: rti.connextdds.LocatorVector) -> None`

Create a copy from another LocatorVector.

4. `__init__(self: rti.connextdds.LocatorVector, arg0: Iterable) -> None`

5. `__init__(self: rti.connextdds.LocatorVector, list: rti.connextdds.LocatorSeq) -> None`

Create a LocatorVector from a list of values.

`__iter__(self: rti.connextdds.LocatorVector) -> Iterator`

Iterate over the contents of the vector.

`__len__(self: rti.connextdds.LocatorVector) -> int`

Get the length of the LocatorVector.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.LocatorVector, arg0: rti.connextdds.LocatorVector) -> bool`

Compare LocatorVectors for inequality.

`__setitem__(self: rti.connextdds.LocatorVector, arg0: int, arg1: rti.connextdds.Locator) -> None`

Set the value at the specified index.

`clear(self: rti.connextdds.LocatorVector) -> None`

Resize LocatorVector to 0.

`resize(self: rti.connextdds.LocatorVector, size: int) -> None`

Resize LocatorVector.

class `rti.connextdds.LogCategory`

Bases: `pybind11_object`

class `LogCategory`

Bases: `pybind11_object`

Members:

`platform` : Log messages pertaining to the underlying platform (hardware and OS) on which RTI Connex is running are in this category.

`communication` : Log messages pertaining to data serialization and deserialization and network traffic are in this category.

`database` : Log messages pertaining to the internal database in which RTI Connex objects are stored are in this category.

`entities` : Log messages pertaining to local and remote entities and to the discovery process are in this category.

`api` : Log messages pertaining to the API layer of RTI Connex (such as method argument validation) are in this category.

discovery : Log messages pertaining to discovery are in this category.

security : Log messages pertaining to Security Plugins are in this category.

user : Log messages that are generated by the user via the public log APIs are in this category.

all_categories : Log messages pertaining to all category of RTI Connex.

`__eq__` (*self: object, other: object*) → bool

`__getstate__` (*self: object*) → int

`__hash__` (*self: object*) → int

`__index__` (*self: rti.connexdds.LogCategory.LogCategory*) → int

`__init__` (*self: rti.connexdds.LogCategory.LogCategory, value: int*) → None

`__int__` (*self: rti.connexdds.LogCategory.LogCategory*) → int

```
__members__ = {'all_categories': <LogCategory.all_categories:
8>, 'api': <LogCategory.api: 4>, 'communication':
<LogCategory.communication: 1>, 'database':
<LogCategory.database: 2>, 'discovery': <LogCategory.discovery:
5>, 'entities': <LogCategory.entities: 3>, 'platform':
<LogCategory.platform: 0>, 'security': <LogCategory.security:
6>, 'user': <LogCategory.user: 7>}
```

`__module__` = 'rti.connexdds'

`__ne__` (*self: object, other: object*) → bool

`__repr__` (*self: object*) → str

`__setstate__` (*self: rti.connexdds.LogCategory.LogCategory, state: int*) → None

`__str__` ()

name(*self: handle*) -> str

`all_categories` = <LogCategory.all_categories: 8>

`api` = <LogCategory.api: 4>

`communication` = <LogCategory.communication: 1>

`database` = <LogCategory.database: 2>

`discovery` = <LogCategory.discovery: 5>

`entities` = <LogCategory.entities: 3>

property name

`platform` = <LogCategory.platform: 0>

security = <LogCategory.security: 6>

user = <LogCategory.user: 7>

property value

__eq__ (*self*: rti.connexdds.LogCategory, *arg0*: rti.connexdds.LogCategory) → bool
Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.LogCategory, *arg0*: rti.connexdds.LogCategory) → bool
Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.LogCategory, *arg0*: rti.connexdds.LogCategory) → bool
Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.LogCategory) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.LogCategory, *arg0*: rti.connexdds.LogCategory.LogCategory)
-> None

Copy constructor.

__int__ (*self*: rti.connexdds.LogCategory) → *rti.connexdds.LogCategory.LogCategory*
Int conversion.

__le__ (*self*: rti.connexdds.LogCategory, *arg0*: rti.connexdds.LogCategory) → bool
Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.LogCategory, *arg0*: rti.connexdds.LogCategory) → bool
Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.LogCategory, *arg0*: rti.connexdds.LogCategory) → bool
Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.LogCategory) → str
String conversion.

all_categories = <LogCategory.all_categories: 8>

api = <LogCategory.api: 4>

communication = <LogCategory.communication: 1>

database = <LogCategory.database: 2>

discovery = <LogCategory.discovery: 5>

entities = <LogCategory.entities: 3>

platform = <LogCategory.platform: 0>

security = <LogCategory.security: 6>

property underlying

Retrieves the actual enumerated value.

user = <LogCategory.user: 7>

class rti.connexdds.Logger

Bases: pybind11_object

__init__ (*args, **kwargs)

__module__ = 'rti.connexdds'

alert (*self*: rti.connexdds.Logger, *msg*: str) → None

Logs an alert message in the user category.

critical (*self*: rti.connexdds.Logger, *msg*: str) → None

Logs a critical message in the user category.

debug (*self*: rti.connexdds.Logger, *msg*: str) → None

Logs a debug message in the user category.

emergency (*self*: rti.connexdds.Logger, *msg*: str) → None

Logs an emergency message in the user category.

error (*self*: rti.connexdds.Logger, *msg*: str) → None

Logs an error message in the user category.

informational (*self*: rti.connexdds.Logger, *msg*: str) → None

Logs an informational message in the user category.

instance = <rti.connexdds.Logger object>

notice (*self*: rti.connexdds.Logger, *msg*: str) → None

Logs a notice message in the user category.

output_file (*self*: rti.connexdds.Logger, *file_name*: str) → None

Set the name of the file to which the logged output is redirected.

output_file_set (*self*: rti.connexdds.Logger, *file_prefix*: str, *file_suffix*: str, *max_bytes*: int, *max_files*: int = LENGTH_UNLIMITED) → None

Set the name of the file to which the logged output is redirected.

output_handler (*self*: rti.connexdds.Logger, *callable_handler*: object) → None

Assigns a callable to which log messages (strings) are directed

property print_format

The message format that RTI Connex will use to log diagnostic information.

reset_output_handler (*self*: rti.connexdds.Logger) → None

Removes the current log handler and sends logging back to the standard output.

property verbosity

The verbosity at which RTI Connex is currently logging diagnostic information.

verbosity_by_category (**args*, ***kwargs*)

Overloaded function.

1. `verbosity_by_category(self: rti.connexdds.Logger, category: rti.connexdds.LogCategory)`
-> rti.connexdds.Verbosity

Get the verbosity at which RTI Connex is currently logging diagnostic information in the given category.

2. `verbosity_by_category(self: rti.connexdds.Logger, category: rti.connexdds.LogCategory, verbosity: rti.connexdds.Verbosity)` -> None

Set the verbosity at which RTI Connex will log diagnostic information in the given category.

warning (*self*: rti.connexdds.Logger, *msg*: str) → None

Logs a warning message in the user category.

class rti.connexdds.LongDouble

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.LongDouble, *arg0*: rti.connexdds.LongDouble) → bool

Test for equality

__getitem__ (*self*: rti.connexdds.LongDouble, *arg0*: int) → int

Get a byte from the LongDouble buffer.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.LongDouble)` -> None

Creates a LongDouble with value 0.

2. `__init__(self: rti.connexdds.LongDouble, byte_sequence: rti.connexdds.Uint8Seq)` -> None

Creates a LongDouble from a sequence of bytes.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.LongDouble, *arg0*: rti.connexdds.LongDouble) → bool

Test for inequality

__setitem__ (*self*: rti.connexdds.LongDouble, *arg0*: int, *arg1*: int) → None

Set a byte in the LongDouble buffer.

__str__ (*self*: rti.connexdds.LongDouble) → str

rti.connexdds.**LongDoubleType**

alias of *Float128Type*

rti.connexdds.**LongLongSeq**

alias of *Int64Seq*

rti.connexdds.**LongLongType**

alias of *Int64Type*

rti.connexdds.**LongSeq**

alias of *Int32Seq*

rti.connexdds.**LongType**

alias of *Int32Type*

class rti.connexdds.**Member**

Bases: pybind11_object

INVALID_ID = 2147483647

__eq__ (*self*: rti.connexdds.Member, *arg0*: rti.connexdds.Member) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.Member, *name*: str, *data_type*: rti.connexdds.DynamicType) → None
2. **__init__**(*self*: rti.connexdds.Member, *name*: str, *data_type*: rti.connexdds.DynamicType, *id*: int = -1, *is_key*: bool = False, *is_optional*: bool = False, *is_pointer*: bool = False) → None

Create a Member with specified properties

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.Member, *arg0*: rti.connexdds.Member) → bool

Test for inequality.

property **bitset**

Checks if member is a bitset

property **id**

The member ID.

property **is_key**

Member key field status.

property name

The member name.

property optional

Member optional status

property pointer

Member pointer status

set_name (*self*: rti.connexdds.Member, *arg0*: str) → rti.connexdds.Member

Set the member name.

property type

Gets the member type.

class rti.connexdds.MemberSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.MemberSeq, *arg0*: rti.connexdds.MemberSeq) → rti.connexdds.MemberSeq

__bool__ (*self*: rti.connexdds.MemberSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.MemberSeq, *x*: rti.connexdds.Member) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.MemberSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.MemberSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.MemberSeq, *arg0*: rti.connexdds.MemberSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.MemberSeq, *s*: slice) -> rti.connexdds.MemberSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.MemberSeq, *arg0*: int) -> rti.connexdds.Member

__hash__ = None

__iadd__ (*self*: rti.connexdds.MemberSeq, *arg0*: rti.connexdds.MemberSeq) → rti.connexdds.MemberSeq

__imul__ (*self*: rti.connexdds.MemberSeq, *arg0*: int) → *rti.connexdds.MemberSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.MemberSeq) -> None
2. **__init__**(*self*: rti.connexdds.MemberSeq, *arg0*: rti.connexdds.MemberSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.MemberSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.MemberSeq) → Iterator

__len__ (*self*: rti.connexdds.MemberSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.MemberSeq, *arg0*: int) → *rti.connexdds.MemberSeq*

__ne__ (*self*: rti.connexdds.MemberSeq, *arg0*: rti.connexdds.MemberSeq) → bool

__rmul__ (*self*: rti.connexdds.MemberSeq, *arg0*: int) → *rti.connexdds.MemberSeq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.MemberSeq, *arg0*: int, *arg1*: rti.connexdds.Member) -> None
2. **__setitem__**(*self*: rti.connexdds.MemberSeq, *arg0*: slice, *arg1*: rti.connexdds.MemberSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.MemberSeq, *x*: rti.connexdds.Member) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.MemberSeq) → None

Clear the contents

count (*self*: rti.connexdds.MemberSeq, *x*: rti.connexdds.Member) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.MemberSeq, *L*: rti.connexdds.MemberSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.MemberSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.MemberSeq, *i*: int, *x*: rti.connexdds.Member) → None
 Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.MemberSeq) -> rti.connexdds.Member

Remove and return the last item

2. pop(*self*: rti.connexdds.MemberSeq, *i*: int) -> rti.connexdds.Member

Remove and return the item at index *i*

remove (*self*: rti.connexdds.MemberSeq, *x*: rti.connexdds.Member) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.**Monitoring**

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.Monitoring, *arg0*: rti.connexdds.Monitoring) → bool
 Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.Monitoring) → None
 Creates the default policy (monitoring is disabled).

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.Monitoring, *arg0*: rti.connexdds.Monitoring) → bool
 Test for inequality.

property application_name

The name of the resource that represents this Connex application.

disabled = <rti.connexdds.Monitoring object>

property distribution_settings

Configures the distribution of telemetry data.

property enable

Enables the collection and distribution of telemetry data for a Connex application using the Monitoring Library 2.0.

enabled = <rti.connexdds.Monitoring object>

property telemetry_data

Configures the collection of telemetry data.

class rti.connexdds.**MonitoringDedicatedParticipantSettings**

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.MonitoringDedicatedParticipantSettings, *arg0*: rti.connextdds.MonitoringDedicatedParticipantSettings) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.MonitoringDedicatedParticipantSettings) → None

Creates an instance with the default settings.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.MonitoringDedicatedParticipantSettings, *arg0*: rti.connextdds.MonitoringDedicatedParticipantSettings) → bool

Test for inequality.

property collector_initial_peers

Determines the initial list of peers that the discovery process will contact to send announcements about the presence of the dedicated_participant.

property domain_id

The domain ID used in the creation of the Monitoring Library 2.0 DomainParticipant.

property enable

Enables the use of a dedicated DomainParticipant to distribute the Connext application telemetry data.

property participant_qos_profile_name

The fully qualified name of the profile used to configure the DomainParticipant that will be used to distribute telemetry data.

class rti.connextdds.MonitoringDistributionSettings

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.MonitoringDistributionSettings, *arg0*: rti.connextdds.MonitoringDistributionSettings) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.MonitoringDistributionSettings) → None

Creates an instance with the default settings.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.MonitoringDistributionSettings, *arg0*: rti.connextdds.MonitoringDistributionSettings) → bool

Test for inequality.

property dedicated_participant

Configures the use of a dedicated DomainParticipant to distribute the Connext application telemetry data.

property event_settings

Configures the distribution of event metrics.

property logging_settings

Configures the distribution of logging messages.

property periodic_settings

Configures the distribution of periodic metrics.

property publisher_qos_profile_name

The fully qualified name of the profile used to configure the Publishers that distribute telemetry data.

class rti.connextdds.MonitoringEventDistributionSettings

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.MonitoringEventDistributionSettings, *arg0*: rti.connextdds.MonitoringEventDistributionSettings) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.MonitoringEventDistributionSettings) → None

Creates an instance with the default settings.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.MonitoringEventDistributionSettings, *arg0*: rti.connextdds.MonitoringEventDistributionSettings) → bool

Test for inequality.

property concurrency_level

Defines how concurrent the push of event metrics to the Monitoring Library 2.0 is

property datawriter_qos_profile_name

The fully qualified name of the profile used to configure the DataWriter that distributes event metrics.

property publication_period

Period at which the event metric thread publishes the event metrics that have changed since the last time they were published.

property thread

The settings of the event metric thread.

class rti.connextdds.MonitoringLoggingDistributionSettings

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.MonitoringLoggingDistributionSettings, *arg0*: rti.connextdds.MonitoringLoggingDistributionSettings) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.MonitoringLoggingDistributionSettings) → None

Creates an instance with the default settings.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.MonitoringLoggingDistributionSettings, *arg0*: rti.connexdds.MonitoringLoggingDistributionSettings) → bool

Test for inequality.

property concurrency_level

Defines how concurrent the push of log messages to the Monitoring Library 2.0 is.

property datawriter_qos_profile_name

The fully qualified name of the profile used to configure the DataWriter that distributes log messages.

property max_historical_logs

The number of log messages that the Monitoring Library 2.0 will keep as history.

property publication_period

Period at which the logging thread publishes log messages.

property thread

The settings of the logging thread.

class rti.connexdds.MonitoringLoggingForwardingSettings

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.MonitoringLoggingForwardingSettings, *arg0*: rti.connexdds.MonitoringLoggingForwardingSettings) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.MonitoringLoggingForwardingSettings) → None

Creates an instance with the default settings.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.MonitoringLoggingForwardingSettings, *arg0*: rti.connexdds.MonitoringLoggingForwardingSettings) → bool

Test for inequality.

property middleware_forwarding_level

The forwarding log level for LogFacility.MIDDLEWARE

property security_event_forwarding_level

The forwarding log level for LogFacility.SECURITY_EVENT

property service_forwarding_level

The forwarding log level for LogFacility.SERVICE

property user_forwarding_level

The forwarding log level for LogFacility.USER

class rti.connexdds.MonitoringMetricSelection

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.MonitoringMetricSelection, *arg0*: rti.connexdds.MonitoringMetricSelection) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.MonitoringMetricSelection) → None

Creates an instance with the default settings.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.MonitoringMetricSelection, *arg0*: rti.connexdds.MonitoringMetricSelection) → bool

Test for inequality.

property disabled_metrics_selection

A sequence of POSIX fnmatch patterns that match the names of the metrics that should not be collected and distributed for the observable resources selected by resource_selection.

property enabled_metrics_selection

A sequence of POSIX fnmatch patterns that match the names of the metrics that should be collected and distributed for the observable resources selected by resource_selection

property resource_selection

An expression pattern that selects a subset of resources by matching the pattern to the resource names. enabled_metrics_selection and disabled_metrics_selection are applied to this subset of resources.

class rti.connexdds.MonitoringMetricSelectionSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.MonitoringMetricSelectionSeq, *arg0*: rti.connexdds.MonitoringMetricSelectionSeq) → rti.connexdds.MonitoringMetricSelectionSeq

__bool__ (*self*: rti.connexdds.MonitoringMetricSelectionSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.MonitoringMetricSelectionSeq, *x*: rti.connexdds.MonitoringMetricSelection) → bool

Return true the container contains x

`__delitem__` (*args, **kwargs)

Overloaded function.

1. `__delitem__(self: rti.connexdds.MonitoringMetricSelectionSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connexdds.MonitoringMetricSelectionSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (self: rti.connexdds.MonitoringMetricSelectionSeq, arg0: rti.connexdds.MonitoringMetricSelectionSeq) → bool

`__getitem__` (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connexdds.MonitoringMetricSelectionSeq, s: slice) -> rti.connexdds.MonitoringMetricSelectionSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.MonitoringMetricSelectionSeq, arg0: int) -> rti.connexdds.MonitoringMetricSelection`

`__hash__` = None

`__iadd__` (self: rti.connexdds.MonitoringMetricSelectionSeq, arg0: rti.connexdds.MonitoringMetricSelectionSeq) → rti.connexdds.MonitoringMetricSelectionSeq

`__imul__` (self: rti.connexdds.MonitoringMetricSelectionSeq, arg0: int) → rti.connexdds.MonitoringMetricSelectionSeq

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.MonitoringMetricSelectionSeq) -> None`
2. `__init__(self: rti.connexdds.MonitoringMetricSelectionSeq, arg0: rti.connexdds.MonitoringMetricSelectionSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.MonitoringMetricSelectionSeq, arg0: Iterable) -> None`

`__iter__` (self: rti.connexdds.MonitoringMetricSelectionSeq) → Iterator

`__len__` (self: rti.connexdds.MonitoringMetricSelectionSeq) → int

`__module__` = 'rti.connexdds'

`__mul__` (self: rti.connexdds.MonitoringMetricSelectionSeq, arg0: int) → rti.connexdds.MonitoringMetricSelectionSeq

__ne__ (*self*: rti.connexdds.MonitoringMetricSelectionSeq, *arg0*: rti.connexdds.MonitoringMetricSelectionSeq) → bool

__rmul__ (*self*: rti.connexdds.MonitoringMetricSelectionSeq, *arg0*: int) → rti.connexdds.MonitoringMetricSelectionSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.MonitoringMetricSelectionSeq, *arg0*: int, *arg1*: rti.connexdds.MonitoringMetricSelection) → None
2. **__setitem__**(*self*: rti.connexdds.MonitoringMetricSelectionSeq, *arg0*: slice, *arg1*: rti.connexdds.MonitoringMetricSelectionSeq) → None

Assign list elements using a slice object

append (*self*: rti.connexdds.MonitoringMetricSelectionSeq, *x*: rti.connexdds.MonitoringMetricSelection) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.MonitoringMetricSelectionSeq) → None

Clear the contents

count (*self*: rti.connexdds.MonitoringMetricSelectionSeq, *x*: rti.connexdds.MonitoringMetricSelection) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.MonitoringMetricSelectionSeq, *L*: rti.connexdds.MonitoringMetricSelectionSeq) → None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.MonitoringMetricSelectionSeq, *L*: Iterable) → None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.MonitoringMetricSelectionSeq, *i*: int, *x*: rti.connexdds.MonitoringMetricSelection) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.MonitoringMetricSelectionSeq) → rti.connexdds.MonitoringMetricSelection

Remove and return the last item

2. **pop**(*self*: rti.connexdds.MonitoringMetricSelectionSeq, *i*: int) → rti.connexdds.MonitoringMetricSelection

Remove and return the item at index *i*

remove (*self*: rti.connexdds.MonitoringMetricSelectionSeq, *x*: rti.connexdds.MonitoringMetricSelection) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.MonitoringPeriodicDistributionSettings

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.MonitoringPeriodicDistributionSettings, *arg0*: rti.connexdds.MonitoringPeriodicDistributionSettings) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.MonitoringPeriodicDistributionSettings) → None

Creates an instance with the default settings.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.MonitoringPeriodicDistributionSettings, *arg0*: rti.connexdds.MonitoringPeriodicDistributionSettings) → bool

Test for inequality.

property datawriter_qos_profile_name

The fully qualified name of the profile used to configure the DataWriter that distributes periodic metrics.

property polling_period

Period at which the periodic metric thread polls and publishes the periodic metrics.

property thread

The settings of the periodic metric thread.

class rti.connexdds.MonitoringTelemetryData

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.MonitoringTelemetryData, *arg0*: rti.connexdds.MonitoringTelemetryData) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.MonitoringTelemetryData) → None

Creates an instance with the default settings.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.MonitoringTelemetryData, *arg0*: rti.connexdds.MonitoringTelemetryData) → bool

Test for inequality.

property logs

The MonitoringLoggingForwardingSettings containing the SyslogVerbosity levels that will be forwarded for the different LogFacility.

property metrics

The sequence of MonitoringMetricSelection containing the event and periodic metrics that will be collected and distributed for a given set of observable resources.

class rti.connexdds.MultiChannel

Bases: pybind11_object

SQL_FILTER_NAME = 'DDSSQL'

STRINGMATCH_FILTER_NAME = 'DDSSTRINGMATCH'

__eq__ (*self*: rti.connexdds.MultiChannel, *arg0*: rti.connexdds.MultiChannel) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.MultiChannel) -> None

Creates the default policy.

2. **__init__**(*self*: rti.connexdds.MultiChannel, *channels*: rti.connexdds.ChannelSettingsSeq, *filter_name*: str = Filter.stringmatch_filter_name) -> None

Creates an instance with the specified channels and filter name.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.MultiChannel, *arg0*: rti.connexdds.MultiChannel) → bool

Test for inequality.

property channels

A sequence of ChannelSettings used to configure the channels' properties. If the length of the sequence is zero, the QoS policy will be ignored.

This property's getter returns a deep copy.

property filter_name

Name of the filter class used to describe the filter expressions of a MultiChannel DataWriter.

class rti.connexdds.MulticastMapping

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.MulticastMapping, *arg0*: rti.connexdds.MulticastMapping) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connextdds.MulticastMapping) -> None

Creates the default mapping

2. **__init__**(self: rti.connextdds.MulticastMapping, addresses: str, topic_expression: str, mapping_function: rti.connextdds.TransportMulticastMappingFunction) -> None

Creates a mapping with the specified parameters

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.MulticastMapping, arg0: rti.connextdds.MulticastMapping) → bool

Test for inequality.

property addresses

Multicast mapping addresses.

property mapping_function

The mapping function.

property topic_expression

The topic expression.

class rti.connextdds.MulticastMappingSeq

Bases: pybind11_object

__add__ (self: rti.connextdds.MulticastMappingSeq, arg0: rti.connextdds.MulticastMappingSeq) → rti.connextdds.MulticastMappingSeq

__bool__ (self: rti.connextdds.MulticastMappingSeq) → bool

Check whether the list is nonempty

__contains__ (self: rti.connextdds.MulticastMappingSeq, x: rti.connextdds.MulticastMapping) → bool

Return true the container contains x

__delitem__ (*args, **kwargs)

Overloaded function.

1. **__delitem__**(self: rti.connextdds.MulticastMappingSeq, arg0: int) -> None

Delete the list elements at index i

2. **__delitem__**(self: rti.connextdds.MulticastMappingSeq, arg0: slice) -> None

Delete list elements using a slice object

__eq__ (self: rti.connextdds.MulticastMappingSeq, arg0: rti.connextdds.MulticastMappingSeq) → bool

__getitem__ (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connexdds.MulticastMappingSeq, s: slice) -> rti.connexdds.MulticastMappingSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.MulticastMappingSeq, arg0: int) -> rti.connexdds.MulticastMapping`

__hash__ = None

__iadd__ (self: rti.connexdds.MulticastMappingSeq, arg0: rti.connexdds.MulticastMappingSeq) → rti.connexdds.MulticastMappingSeq

__imul__ (self: rti.connexdds.MulticastMappingSeq, arg0: int) → rti.connexdds.MulticastMappingSeq

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.MulticastMappingSeq) -> None`
2. `__init__(self: rti.connexdds.MulticastMappingSeq, arg0: rti.connexdds.MulticastMappingSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.MulticastMappingSeq, arg0: Iterable) -> None`

__iter__ (self: rti.connexdds.MulticastMappingSeq) → Iterator

__len__ (self: rti.connexdds.MulticastMappingSeq) → int

__module__ = 'rti.connexdds'

__mul__ (self: rti.connexdds.MulticastMappingSeq, arg0: int) → rti.connexdds.MulticastMappingSeq

__ne__ (self: rti.connexdds.MulticastMappingSeq, arg0: rti.connexdds.MulticastMappingSeq) → bool

__rmul__ (self: rti.connexdds.MulticastMappingSeq, arg0: int) → rti.connexdds.MulticastMappingSeq

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connexdds.MulticastMappingSeq, arg0: int, arg1: rti.connexdds.MulticastMapping) -> None`
2. `__setitem__(self: rti.connexdds.MulticastMappingSeq, arg0: slice, arg1: rti.connexdds.MulticastMappingSeq) -> None`

Assign list elements using a slice object

append (*self*: rti.connexdds.MulticastMappingSeq, *x*: rti.connexdds.MulticastMapping) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.MulticastMappingSeq) → None

Clear the contents

count (*self*: rti.connexdds.MulticastMappingSeq, *x*: rti.connexdds.MulticastMapping) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connexdds.MulticastMappingSeq, *L*: rti.connexdds.MulticastMappingSeq) → None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connexdds.MulticastMappingSeq, *L*: Iterable) → None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.MulticastMappingSeq, *i*: int, *x*: rti.connexdds.MulticastMapping) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.MulticastMappingSeq) → rti.connexdds.MulticastMapping

Remove and return the last item

2. pop(*self*: rti.connexdds.MulticastMappingSeq, *i*: int) → rti.connexdds.MulticastMapping

Remove and return the item at index *i*

remove (*self*: rti.connexdds.MulticastMappingSeq, *x*: rti.connexdds.MulticastMapping) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.NoOpAnyDataReaderListener

Bases: *AnyDataReaderListener*

__init__ (*self*: rti.connexdds.NoOpAnyDataReaderListener) → None

__module__ = 'rti.connexdds'

on_data_available (*self*: rti.connexdds.NoOpAnyDataReaderListener, *arg0*: rti.connexdds.AnyDataReader) → None

Data available callback.

on_liveliness_changed (*self*: rti.connexdds.NoOpAnyDataReaderListener, *arg0*: rti.connexdds.AnyDataReader, *arg1*: rti.connexdds.LivelinessChangedStatus) → None

Liveliness changed callback.

```
on_requested_deadline_missed (self: rti.connexdds.NoOpAnyDataReaderListener, arg0:
    rti.connexdds.AnyDataReader, arg1:
    rti.connexdds.RequestedDeadlineMissedStatus) → None
```

Requested deadline missed callback.

```
on_requested_incompatible_qos (self: rti.connexdds.NoOpAnyDataReaderListener, arg0:
    rti.connexdds.AnyDataReader, arg1:
    rti.connexdds.RequestedIncompatibleQosStatus) →
    None
```

Requested incompatible QoS callback.

```
on_sample_lost (self: rti.connexdds.NoOpAnyDataReaderListener, arg0:
    rti.connexdds.AnyDataReader, arg1: rti.connexdds.SampleLostStatus) →
    None
```

Sample lost callback.

```
on_sample_rejected (self: rti.connexdds.NoOpAnyDataReaderListener, arg0:
    rti.connexdds.AnyDataReader, arg1:
    rti.connexdds.SampleRejectedStatus) → None
```

Sample rejected callback.

```
on_subscription_matched (self: rti.connexdds.NoOpAnyDataReaderListener, arg0:
    rti.connexdds.AnyDataReader, arg1:
    rti.connexdds.SubscriptionMatchedStatus) → None
```

Subscription matched callback.

```
class rti.connexdds.NoOpAnyDataWriterListener
```

```
    Bases: AnyDataWriterListener
```

```
    __init__ (self: rti.connexdds.NoOpAnyDataWriterListener) → None
```

```
    __module__ = 'rti.connexdds'
```

```
on_application_acknowledgment (self: rti.connexdds.NoOpAnyDataWriterListener, arg0:
    rti.connexdds.AnyDataWriter, arg1:
    rti.connexdds.AcknowledgmentInfo) → None
```

On application acknowledgment callback

```
on_instance_replaced (self: rti.connexdds.NoOpAnyDataWriterListener, arg0:
    rti.connexdds.AnyDataWriter, arg1: rti.connexdds.InstanceHandle)
    → None
```

On instance replaced callback.

```
on_liveliness_lost (self: rti.connexdds.NoOpAnyDataWriterListener, arg0:
    rti.connexdds.AnyDataWriter, arg1:
    rti.connexdds.LivelinessLostStatus) → None
```

Liveliness lost callback.

on_offered_deadline_missed (*self*: rti.connexdds.NoOpAnyDataWriterListener, *arg0*: rti.connexdds.AnyDataWriter, *arg1*: rti.connexdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

on_offered_incompatible_qos (*self*: rti.connexdds.NoOpAnyDataWriterListener, *arg0*: rti.connexdds.AnyDataWriter, *arg1*: rti.connexdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

on_publication_matched (*self*: rti.connexdds.NoOpAnyDataWriterListener, *arg0*: rti.connexdds.AnyDataWriter, *arg1*: rti.connexdds.PublicationMatchedStatus) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connexdds.NoOpAnyDataWriterListener, *arg0*: rti.connexdds.AnyDataWriter, *arg1*: rti.connexdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connexdds.NoOpAnyDataWriterListener, *arg0*: rti.connexdds.AnyDataWriter, *arg1*: rti.connexdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connexdds.NoOpAnyDataWriterListener, *arg0*: rti.connexdds.AnyDataWriter, *arg1*: rti.connexdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class rti.connexdds.NoOpAnyTopicListener

Bases: *AnyTopicListener*

__init__ (*self*: rti.connexdds.NoOpAnyTopicListener) → None

__module__ = 'rti.connexdds'

on_inconsistent_topic (*self*: rti.connexdds.NoOpAnyTopicListener, *arg0*: rti.connexdds.AnyTopic, *arg1*: rti.connexdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class rti.connexdds.NoOpDataReaderListener

Bases: *DataReaderListener*

__init__ (*self*: rti.connexdds.NoOpDataReaderListener) → None

```
__module__ = 'rti.connexdds'
```

```
on_data_available (self: rti.connexdds.NoOpDataReaderListener, arg0:  
rti.connexdds.DataReader) → None
```

Data available callback.

```
on_liveliness_changed (self: rti.connexdds.NoOpDataReaderListener, arg0:  
rti.connexdds.DataReader, arg1:  
rti.connexdds.LivelinessChangedStatus) → None
```

Liveliness changed callback.

```
on_requested_deadline_missed (self: rti.connexdds.NoOpDataReaderListener, arg0:  
rti.connexdds.DataReader, arg1:  
rti.connexdds.RequestedDeadlineMissedStatus) → None
```

Requested deadline missed callback.

```
on_requested_incompatible_qos (self: rti.connexdds.NoOpDataReaderListener, arg0:  
rti.connexdds.DataReader, arg1:  
rti.connexdds.RequestedIncompatibleQosStatus) →  
None
```

Requested incompatible QoS callback.

```
on_sample_lost (self: rti.connexdds.NoOpDataReaderListener, arg0:  
rti.connexdds.DataReader, arg1: rti.connexdds.SampleLostStatus) → None
```

Sample lost callback.

```
on_sample_rejected (self: rti.connexdds.NoOpDataReaderListener, arg0:  
rti.connexdds.DataReader, arg1: rti.connexdds.SampleRejectedStatus)  
→ None
```

Sample rejected callback.

```
on_subscription_matched (self: rti.connexdds.NoOpDataReaderListener, arg0:  
rti.connexdds.DataReader, arg1:  
rti.connexdds.SubscriptionMatchedStatus) → None
```

Subscription matched callback.

```
class rti.connexdds.NoOpDataWriterListener
```

```
Bases: DataWriterListener
```

```
__init__ (self: rti.connexdds.NoOpDataWriterListener) → None
```

```
__module__ = 'rti.connexdds'
```

```
on_application_acknowledgment (self: rti.connexdds.NoOpDataWriterListener, arg0:  
rti.connexdds.DataWriter, arg1:  
rti.connexdds.AcknowledgmentInfo) → None
```

On application acknowledgment callback

```
on_instance_replaced (self: rti.connexdds.NoOpDataWriterListener, arg0:  
rti.connexdds.DataWriter, arg1: rti.connexdds.InstanceHandle) →  
None
```

On instance replaced callback.

```
on_liveliness_lost (self: rti.connextdds.NoOpDataWriterListener, arg0:
                    rti.connextdds.DataWriter, arg1: rti.connextdds.LivelinessLostStatus) →
                    None
```

Liveliness lost callback.

```
on_offered_deadline_missed (self: rti.connextdds.NoOpDataWriterListener, arg0:
                             rti.connextdds.DataWriter, arg1:
                             rti.connextdds.OfferedDeadlineMissedStatus) → None
```

Offered deadline missed callback.

```
on_offered_incompatible_qos (self: rti.connextdds.NoOpDataWriterListener, arg0:
                               rti.connextdds.DataWriter, arg1:
                               rti.connextdds.OfferedIncompatibleQosStatus) → None
```

Offered incompatible QoS callback.

```
on_publication_matched (self: rti.connextdds.NoOpDataWriterListener, arg0:
                          rti.connextdds.DataWriter, arg1:
                          rti.connextdds.PublicationMatchedStatus) → None
```

Publication matched callback.

```
on_reliable_reader_activity_changed (self: rti.connextdds.NoOpDataWriterListener,
                                       arg0: rti.connextdds.DataWriter, arg1:
                                       rti.connextdds.ReliableReaderActivityChanged-
                                       Status) →
                                       None
```

Reliable reader activity changed callback.

```
on_reliable_writer_cache_changed (self: rti.connextdds.NoOpDataWriterListener, arg0:
                                    rti.connextdds.DataWriter, arg1:
                                    rti.connextdds.ReliableWriterCacheChangedStatus)
→ None
```

Reliable writer cache changed callback.

```
on_service_request_accepted (self: rti.connextdds.NoOpDataWriterListener, arg0:
                               rti.connextdds.DataWriter, arg1:
                               rti.connextdds.ServiceRequestAcceptedStatus) → None
```

On service request accepted callback.

```
class rti.connextdds.NoOpDomainParticipantListener
    Bases: DomainParticipantListener
    __init__ (self: rti.connextdds.NoOpDomainParticipantListener) → None
    __module__ = 'rti.connextdds'
```

on_invalid_local_identity_status_advance_notice (*self*: rti.connextdds.NoOpDomainParticipantListener, *arg0*: rti.connextdds.DomainParticipant, *arg1*: rti.connextdds.InvalidLocalIdentityAdvanceNoticeStatus) → None

On invalid local identity status advance notice callback

```
class rti.connextdds.NoOpPublisherListener
```

Bases: *PublisherListener*

```
__init__ (self: rti.connextdds.NoOpPublisherListener) → None
```

```
__module__ = 'rti.connextdds'
```

```
class rti.connextdds.NoOpSubscriberListener
```

Bases: *SubscriberListener*

```
__init__ (self: rti.connextdds.NoOpSubscriberListener) → None
```

```
__module__ = 'rti.connextdds'
```

```
on_data_on_readers (self: rti.connextdds.NoOpSubscriberListener, arg0: rti.connextdds.Subscriber) → None
```

Data on datareaders callback.

```
class rti.connextdds.NoOpTopicListener
```

Bases: *TopicListener*

```
__init__ (self: rti.connextdds.NoOpTopicListener) → None
```

```
__module__ = 'rti.connextdds'
```

```
on_inconsistent_topic (self: rti.connextdds.NoOpTopicListener, arg0: rti.connextdds.Topic, arg1: rti.connextdds.InconsistentTopicStatus) → None
```

Inconsistent topic callback.

```
exception rti.connextdds.NotAllowedBySecurityError
```

Bases: *Exception*

```
__module__ = 'rti.connextdds'
```

```
exception rti.connextdds.NotEnabledError
```

Bases: *Exception*

```
__module__ = 'rti.connextdds'
```

```
exception rti.connextdds.NullReferenceError
```

Bases: *Exception*


```

    __module__ = 'rti.connexdds'

rti.connexdds.OctetSeq
    alias of Uint8Seq

rti.connexdds.OctetType
    alias of Uint8Type

class rti.connexdds.OfferedDeadlineMissedStatus
    Bases: pybind11_object

    __init__ (*args, **kwargs)

    __module__ = 'rti.connexdds'

    property last_instance_handle
        Handle to the last instance in the DataWriter for which an offered deadline was missed.

    property total_count
        The count of a DataWriter's failures to meet write deadlines.

    property total_count_change
        The delta in total_count since the last time the listener was called or the status was read.

class rti.connexdds.OfferedIncompatibleQosStatus
    Bases: pybind11_object

    __init__ (*args, **kwargs)

    __module__ = 'rti.connexdds'

    property last_policy
        The policy class of one of the policies that was found to be incompatible the last time an incompatibility was detected.

    property policies
        A list containing for each policy the total number of times that the concerned DataWriter discovered a DataReader for the same Topic and common partition with a requested QoS that is incompatible with that offered by the DataWriter.

    property total_count
        Total number of times the concerned DataWriter discovered a DataReader for the same Topic, common partition with a requested QoS that is incompatible with that offered by the DataWriter.

    property total_count_change
        The delta in total_count since the last time the listener was called or the status was read.

exception rti.connexdds.OutOfResourcesError
    Bases: Exception

    __module__ = 'rti.connexdds'

```

```
class rti.connexdds.Ownership
```

```
Bases: pybind11_object
```

```
__eq__ (self: rti.connexdds.Ownership, arg0: rti.connexdds.Ownership) → bool
```

```
Test for equality.
```

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

```
Overloaded function.
```

```
1. __init__(self: rti.connexdds.Ownership) -> None
```

```
Creates an ownership policy set to shared.
```

```
2. __init__(self: rti.connexdds.Ownership, kind: rti.connexdds.OwnershipKind) -> None
```

```
Creates an instance with the specified ownership kind.
```

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.Ownership, arg0: rti.connexdds.Ownership) → bool
```

```
Test for inequality.
```

```
exclusive = <rti.connexdds.Ownership object>
```

```
property kind
```

```
The ownership kind.
```

```
shared = <rti.connexdds.Ownership object>
```

```
class rti.connexdds.OwnershipKind
```

```
Bases: pybind11_object
```

```
EXCLUSIVE = <OwnershipKind.EXCLUSIVE: 1>
```

```
class OwnershipKind
```

```
Bases: pybind11_object
```

```
Members:
```

```
SHARED : [default] Indicates shared ownership for each instance.
```

Multiple writers are allowed to update the same instance and all the updates are made available to the readers. In other words there is no concept of an owner for the instances.

This is the default behavior if the OWNERSHIP policy is not specified or supported.

```
EXCLUSIVE : Indicates each instance can only be owned by one DataWriter, but the owner of an instance can change dynamically.
```

The selection of the owner is controlled by the setting of the OWNERSHIP_STRENGTH policy. The owner is always set to be the highest-strength DataWriter object among the ones currently active (as determined by the LIVELINESS).

```

EXCLUSIVE = <OwnershipKind.EXCLUSIVE: 1>
SHARED = <OwnershipKind.SHARED: 0>

__eq__(self: object, other: object) → bool
__getstate__(self: object) → int
__hash__(self: object) → int
__index__(self: rti.connextdds.OwnershipKind.OwnershipKind) → int
__init__(self: rti.connextdds.OwnershipKind.OwnershipKind, value: int) → None
__int__(self: rti.connextdds.OwnershipKind.OwnershipKind) → int

__members__ = {'EXCLUSIVE': <OwnershipKind.EXCLUSIVE: 1>,
'SHARED': <OwnershipKind.SHARED: 0>}

__module__ = 'rti.connextdds'

__ne__(self: object, other: object) → bool
__repr__(self: object) → str
__setstate__(self: rti.connextdds.OwnershipKind.OwnershipKind, state: int) → None
__str__()
    name(self: handle) -> str

property name

property value

```

```

SHARED = <OwnershipKind.SHARED: 0>

__eq__(self: rti.connextdds.OwnershipKind, arg0: rti.connextdds.OwnershipKind) → bool
    Apply operator to underlying enumerated values.

__ge__(self: rti.connextdds.OwnershipKind, arg0: rti.connextdds.OwnershipKind) → bool
    Apply operator to underlying enumerated values.

__gt__(self: rti.connextdds.OwnershipKind, arg0: rti.connextdds.OwnershipKind) → bool
    Apply operator to underlying enumerated values.

__hash__ = None

__init__(*args, **kwargs)
    Overloaded function.
    1. __init__(self: rti.connextdds.OwnershipKind) -> None
    Initializes enum to 0.

```

2. `__init__(self: rti.connexdds.OwnershipKind, arg0: rti.connexdds.OwnershipKind.OwnershipKind) -> None`

Copy constructor.

`__int__(self: rti.connexdds.OwnershipKind) -> rti.connexdds.OwnershipKind.OwnershipKind`

Int conversion.

`__le__(self: rti.connexdds.OwnershipKind, arg0: rti.connexdds.OwnershipKind) -> bool`

Apply operator to underlying enumerated values.

`__lt__(self: rti.connexdds.OwnershipKind, arg0: rti.connexdds.OwnershipKind) -> bool`

Apply operator to underlying enumerated values.

`__module__ = 'rti.connexdds'`

`__ne__(self: rti.connexdds.OwnershipKind, arg0: rti.connexdds.OwnershipKind) -> bool`

Apply operator to underlying enumerated values.

`__str__(self: rti.connexdds.OwnershipKind) -> str`

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.OwnershipStrength

Bases: `pybind11_object`

`__eq__(self: rti.connexdds.OwnershipStrength, arg0: rti.connexdds.OwnershipStrength) -> bool`

Test for equality.

`__hash__ = None`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connexdds.OwnershipStrength) -> None`

Creates an instance with the default strength (0).

2. `__init__(self: rti.connexdds.OwnershipStrength, strength: int) -> None`

Creates an instance with the specified strength value.

`__module__ = 'rti.connexdds'`

`__ne__(self: rti.connexdds.OwnershipStrength, arg0: rti.connexdds.OwnershipStrength) -> bool`

Test for inequality.

property value

The ownership kind.

class rti.connexdds.ParticipantBuiltinTopicData

Bases: `pybind11_object`

class ContentFilterBases: *ContentFilterBase***__init__** (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilter) → None**__module__** = 'rti.connexdds'**compile** (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilter, *expression*: str, *parameters*: rti.connexdds.StringSeq, *type_code*: Optional[rti.connexdds.DynamicType], *type_class_name*: str, *old_compile_data*: Optional[object]) → Optional[object]

Compile an instance of the content filter according to the filter expression and parameters of the given data type.

evaluate (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilter, *compile_data*: Optional[object], *sample*: rti.connexdds.ParticipantBuiltinTopicData, *meta_data*: rti.connexdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

finalize (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilter, *compile_data*: Optional[object]) → None

A previously compiled instance of the content filter is no longer in use and resources can now be cleaned up.

class ContentFilteredTopicBases: *ITopicDescription*, *IAnyTopic***__eq__** (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → bool

Test for equality.

__hash__ = None**__init__** (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *topic*: rti.connexdds.ParticipantBuiltinTopicData.Topic, *name*: str, *contentfilter*: rti.connexdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *topic_description*: rti.connexdds.ParticipantBuiltinTopicData.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

__module__ = 'rti.connexdds'**__ne__** (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → bool

Test for inequality.

append_to_expression_parameter (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *index*: int, *extension*: str) → None

Append the extension to the end of parameter at the provided index, separated by a comma.

property filter_expression

Get the filter expression

property filter_parameters

Get/set the filter parameters.

static find (*participant*: rti.connexdds.DomainParticipant, *name*: str) → Optional[rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

remove_from_expression_parameter (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *index*: int, *remove_term*: str) → None

Removes the specified term from the parameter at the provided index.

set_filter (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *arg0*: rti.connexdds.Filter) → None

Set the filter.

property topic

Get the underlying Topic.

class ContentFilteredTopicSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq

__bool__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) -> None

Delete the list elements at index i

2. **__delitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *s*: slice) -> rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) -> rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic

__hash__ = None

__iadd__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → *rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq*

__imul__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) → *rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) -> None

2. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → Iterator

__len__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) → *rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq*

__ne__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → bool

__rmul__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) → *rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int, *arg1*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic) -> None

2. `__setitem__(self: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, arg0: slice, arg1: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) -> None`

Assign list elements using a slice object

append (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) → None

Clear the contents

count (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. `extend(self: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, L: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *i*: int, *x*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq) -> rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic`

Remove and return the last item

2. `pop(self: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, i: int) -> rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic`

Remove and return the item at index *i*

remove (*self*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class DataReader

Bases: *IDataReader*

class Selector

Bases: *pybind11_object*

`__init__` (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector, *datareader*: rti.connexdds.ParticipantBuiltinTopicData.DataReader) → None

Create a Selector for a DataReader to read/take based on selected conditions

__module__ = 'rti.connextdds'

condition (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector, *condition*: rti.connextdds.IReadCondition) → *rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector*

Select samples based on a ReadCondition.

content (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector, *query*: rti.connextdds.Query) → *rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector*

Select samples based on a Query.

instance (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector*

Select a specific instance to read/take.

max_samples (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector, *max*: int) → *rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector*

Limit the number of samples read/taken by the Select.

next_instance (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector, *previous*: rti.connextdds.InstanceHandle) → *rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector*

Select the instance after the specified instance to read/take.

read (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector) → list
Read copies of available samples (data and info) based on the Selector settings.

read_data (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector) → list
Read copies of available valid data based on the Selector settings.

read_loaned (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector) → *rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples*

Take available samples (data and info) based on the Selector settings and return them in a loaned container.

state (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector, *state*: rti.connextdds.DataState) → *rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector*

Select samples with a specified data state.

take (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector) → list
Take copies of available samples (data and info) based on the Selector settings.

take_data (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector) → list
 Take copies of available valid data based on the Selector settings.

take_loaned (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader.Selector) →
rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples

Read available samples (data and info) based on the Selector settings and return them in a loaned container.

__enter__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader) →
rti.connextdds.ParticipantBuiltinTopicData.DataReader

Enter a context for this DataReader, to be cleaned up on exiting context

__eq__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg0*:
 rti.connextdds.ParticipantBuiltinTopicData.DataReader) → bool

Test for equality.

__exit__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg0*: *object*, *arg1*:
object, *arg2*: *object*) → None

Exit the context for this DataReader, cleaning up resources.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *topic*: rti.connextdds.ParticipantBuiltinTopicData.Topic) -> None

Create a DataReader in the implicit subscriber with default QoS.

2. **__init__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *topic*: rti.connextdds.ParticipantBuiltinTopicData.Topic, *qos*: rti.connextdds.DataReaderQos, *listener*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener = None, *mask*: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader in the implicit subscriber with specific QoS and a listener.

3. **__init__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *cft*: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic) -> None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with default QoS.

4. **__init__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *cft*: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic, *qos*: rti.connextdds.DataReaderQos, *listener*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener = None, *mask*: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with specific QoS.

5. **__init__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *subscriber*: rti.connextdds.Subscriber, *topic*: rti.connextdds.ParticipantBuiltinTopicData.Topic) -> None

Create a DataReader.

6. **__init__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *subscriber*: rti.connextdds.Subscriber, *topic*: rti.connextdds.ParticipantBuiltinTopicData.Topic, *qos*: rti.connextdds.DataReaderQos, *listener*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener = None, *mask*: rti.connextdds.StatusMask = StatusMask.ALL)

-> None

Create a DataReader in a subscriber with specific QoS and a listener.

7. `__init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader, subscriber: rti.connextdds.Subscriber, cft: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic)` -> None

Create a DataReader with a ContentFilteredTopic in a subscriber with default QoS.

8. `__init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader, subscriber: rti.connextdds.Subscriber, cft: rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL)` -> None

Create a DataReader with a ContentFilteredTopic in a subscriber with specific QoS.

9. `__init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader, reader: rti.connextdds.IAnyDataReader)` -> None

Get a typed DataReader from an AnyDataReader.

10. `__init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader, entity: rti.connextdds.IEntity)` -> None

Get a typed DataReader from an Entity.

`__lshift__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader, arg0: rti.connextdds.DataReaderQos) → rti.connextdds.ParticipantBuiltinTopicData.DataReader`

Set the DataReaderQos for this DataReader.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader, arg0: rti.connextdds.ParticipantBuiltinTopicData.DataReader) → bool`

Test for inequality.

`__rshift__(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader, arg0: rti.connextdds.DataReaderQos) → rti.connextdds.ParticipantBuiltinTopicData.DataReader`

Get the DataReaderQos from this DataReader

`acknowledge_all(*args, **kwargs)`

Overloaded function.

1. `acknowledge_all(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader)` -> None

Acknowledge all previously accessed samples.

2. `acknowledge_all(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader, arg0: rti.connextdds.AckResponseData)` -> None

Acknowledge all previously accessed samples.

`acknowledge_sample(*args, **kwargs)`

Overloaded function.

1. `acknowledge_sample(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader, sample_info: rti.connextdds.SampleInfo)` -> None

Acknowledge a single sample.

2. `acknowledge_sample(self: rti.connextdds.ParticipantBuiltinTopicData.DataReader, sample_info: rti.connextdds.SampleInfo, ack_response_data: rti.connextdds.AckResponseData) -> None`

Acknowledge a single sample with ack response data.

close (*self*: `rti.connextdds.ParticipantBuiltinTopicData.DataReader`) → None

Close this DataReader.

property datareader_cache_status

Get the DataReaderCacheStatus for the DataReader.

property datareader_protocol_status

Get the DataReaderProtocolStatus for the DataReader.

property default_filter_state

Returns the filter state for the read/take operations.

static find_all_by_topic (*subscriber*: `rti.connextdds.Subscriber`, *topic_name*: `str`) → `rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq`

Retrieve all DataReaders for the given topic name in the subscriber.

static find_by_name (**args*, ***kwargs*)

Overloaded function.

1. `find_by_name(participant: rti.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.ParticipantBuiltinTopicData.DataReader]`

Find DataReader in DomainParticipant with the DataReader's name, returning the first found.

2. `find_by_name(subscriber: rti.connextdds.Subscriber, name: str) -> Optional[rti.connextdds.ParticipantBuiltinTopicData.DataReader]`

Find DataReader in Subscriber with the DataReader's name, returning the first found.

static find_by_topic (*subscriber*: `rti.connextdds.Subscriber`, *name*: `str`) → `Optional[rti.connextdds.ParticipantBuiltinTopicData.DataReader]`

Find DataReader in Subscriber with a topic name, returning the first found.

is_matched_publication_alive (*self*: `rti.connextdds.ParticipantBuiltinTopicData.DataReader`, *arg0*: `rti.connextdds.InstanceHandle`) → bool

Check if a matched publication is alive.

key_value (*self*: `rti.connextdds.ParticipantBuiltinTopicData.DataReader`, *handle*: `rti.connextdds.InstanceHandle`) → `rti.connextdds.ParticipantBuiltinTopicData`

Retrieve the instance key that corresponds to an instance handle.

property listener

Gets or sets the listener with StatusMask.ALL

property liveliness_changed_status

Get the LivelinessChangedStatus for this DataReader.

lookup_instance (*self*: `rti.connextdds.ParticipantBuiltinTopicData.DataReader`, *key_holder*: `rti.connextdds.ParticipantBuiltinTopicData`) → `rti.connextdds.InstanceHandle`

Retrieve the instance handle that corresponds to an instance key_holder

matched_publication_data (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.PublicationBuiltinTopicData*

Get the PublicationBuiltinTopicData for a publication matched to this DataReader.

matched_publication_datareader_protocol_status (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *publication_handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.DataReaderProtocolStatus*

Get the DataReaderProtocolStatus for the DataReader based on the matched publication handle.

matched_publication_participant_data (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a publication matched to this DataReader.

property matched_publications

Get a copy of the list of the currently matched publication handles.

property qos

The DataReaderQos for this DataReader.

This property's getter returns a deep copy.

read (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReader) → list

Read copies of all available samples (data and info).

read_data (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReader) → list

Read copies of all available valid data.

read_loaned (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReader) →

rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples

Read all available samples (data and info) and return them in a loaned container.

property requested_deadline_missed_status

Get the RequestedDeadlineMissed status for the DataReader

property requested_incompatible_qos_status

Get the RequestedIncompatibleQosStatus for the DataReader.

property sample_lost_status

Get the SampleLostStatus for the DataReader.

property sample_rejected_status

Get the SampleRejectedStatus for the DataReader.

select (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader) →
 dds::sub::DataReader<dds::topic::TParticipantBuiltinTopicData<rti::topic::Participant-
 BuiltinTopicDataImpl>,
 rti::sub::DataReaderImpl>::Selector

Get a Selector to perform complex data selections, such as per-instance selection, content, and status filtering.

set_listener (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *listener*:
 rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener, *event_mask*:
 rti.connextdds.StatusMask) → None

Set the listener and associated event mask.

property subscriber

Returns the parent Subscriber of the DataReader.

property subscription_matched_status

Get the SubscriptionMatchedStatus for the DataReader.

take (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader) → list

Take copies of all available samples (data and info).

take_data (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader) → list

Take copies of all available valid data.

take_loaned (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReader) →
rti.connextdds.ParticipantBuiltinTopicData.LoanedSamples

Take all available samples (data and info) and return them in a loaned container.

property topic_description

Returns the TopicDescription associated with the DataReader.

property topic_name

Get the topic name associated with this DataReader.

property type_name

Get the type name associated with this DataReader.

wait_for_historical_data (*self*:
 rti.connextdds.ParticipantBuiltinTopicData.DataReader,
max_wait: rti.connextdds.Duration) → None

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind.

wait_for_historical_data_async (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *max_wait*: rti.connexdds.Duration) → object

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind. This call is awaitable and only for use with asyncio.

class DataReaderListener

Bases: pybind11_object

__init__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener) → None

__module__ = 'rti.connexdds'

on_data_available (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataReader) → None

Data available callback.

on_liveliness_changed (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connexdds.LivelinessChangedStatus) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SampleLostStatus) → None

Sample lost callback.

on_sample_rejected (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class DataReaderSeq

Bases: pybind11_object

__add__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) → *rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq*

__bool__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *x*: rti.connextdds.ParticipantBuiltinTopicData.DataReader) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *arg0*: int) → None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *arg0*: slice) → None

Delete list elements using a slice object

__eq__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *s*: slice) → rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *arg0*: int) → rti.connextdds.ParticipantBuiltinTopicData.DataReader

__hash__ = None

__iadd__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq) → *rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq*

__imul__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq, *arg0*: int) → *rti.connextdds.ParticipantBuiltinTopicData.DataReaderSeq*

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq) -> None
2. **__init__**(self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq) -> None

Copy constructor

3. **__init__**(self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: Iterable) -> None

__iter__ (self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq) → Iterator

__len__ (self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq) → int

__module__ = 'rti.connexdds'

__mul__ (self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: int) → rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq

__ne__ (self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq) → bool

__rmul__ (self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: int) → rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq

__setitem__ (*args, **kwargs)

Overloaded function.

1. **__setitem__**(self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: int, arg1: rti.connexdds.ParticipantBuiltinTopicData.DataReader) -> None
2. **__setitem__**(self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, arg0: slice, arg1: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq) -> None

Assign list elements using a slice object

append (self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, x: rti.connexdds.ParticipantBuiltinTopicData.DataReader) → None

Add an item to the end of the list

clear (self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq) → None

Clear the contents

count (self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, x: rti.connexdds.ParticipantBuiltinTopicData.DataReader) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. **extend**(self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, L: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(self: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, *i*: int, *x*: rti.connexdds.ParticipantBuiltinTopicData.DataReader) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq) → rti.connexdds.ParticipantBuiltinTopicData.DataReader

Remove and return the last item

2. pop(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, *i*: int) → rti.connexdds.ParticipantBuiltinTopicData.DataReader

Remove and return the item at index *i*

remove (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq, *x*: rti.connexdds.ParticipantBuiltinTopicData.DataReader) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class DataWriter

Bases: *IEntity*, *IAnyDataWriter*

__enter__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter) → *rti.connexdds.ParticipantBuiltinTopicData.DataWriter*

Enter a context for this DataWriter, to be cleaned up on exiting context

__eq__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter) → bool

Test for equality.

__exit__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, *arg0*: *object*, *arg1*: *object*, *arg2*: *object*) → None

Exit the context for this DataWriter, cleaning up resources.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, *topic*: rti.connexdds.ParticipantBuiltinTopicData.Topic) → None

Creates a DataWriter in the implicit publisher with default QoS.

2. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, *topic*: rti.connexdds.ParticipantBuiltinTopicData.Topic, *qos*: rti.connexdds.DataWriterQos, *listener*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterListener = None, *mask*: rti.connexdds.StatusMask = StatusMask.ALL) → None

Creates a DataWriter in the implicit publisher with specific QoS and optionally a listener.

3. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, *pub*: rti.connexdds.Publisher, *topic*: rti.connexdds.ParticipantBuiltinTopicData.Topic) → None

Creates a DataWriter in a publisher with default QoS.

4. `__init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.ParticipantBuiltinTopicData.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL)`
 -> None

Creates a DataWriter in a publisher with specific QoS and optionally a listener.

5. `__init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, writer: rti.connextdds.IAnyDataWriter)` -> None

Create a typed DataWriter from an AnyDataWriter.

6. `__init__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, entity: rti.connextdds.IEntity)` -> None

Create a typed DataWriter from an Entity.

`__lshift__` (*args, **kwargs)

Overloaded function.

1. `__lshift__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg0: rti.connextdds.DataWriterQos)` -> `rti.connextdds.ParticipantBuiltinTopicData.DataWriter`

Sets the DataWriterQos.

2. `__lshift__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg0: Tuple[rti.connextdds.ParticipantBuiltinTopicData, rti.connextdds.Time])` -> `rti.connextdds.ParticipantBuiltinTopicData.DataWriter`

Writes a paired sample with a timestamp.

3. `__lshift__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg0: Tuple[rti.connextdds.ParticipantBuiltinTopicData, rti.connextdds.InstanceHandle])` -> `rti.connextdds.ParticipantBuiltinTopicData.DataWriter`

Writes a paired sample with an instance handle.

4. `__lshift__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg0: rti.connextdds.ParticipantBuiltinTopicData.TimestampedSeq)` -> `rti.connextdds.ParticipantBuiltinTopicData.DataWriter`

Writes a sequence of pairs of samples with timestamps.

5. `__lshift__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg0: rti.connextdds.ParticipantBuiltinTopicDataSeq)` -> `rti.connextdds.ParticipantBuiltinTopicData.DataWriter`

Writes a sequence of samples.

6. `__lshift__(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg0: rti.connextdds.ParticipantBuiltinTopicData)` -> `rti.connextdds.ParticipantBuiltinTopicData.DataWriter`

Writes a sample.

`__module__` = 'rti.connextdds'

`__ne__` (self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg0: rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → bool

Test for inequality.

`__rshift__` (self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg0: rti.connextdds.DataWriterQos) → *rti.connextdds.ParticipantBuiltinTopicData.DataWriter*

Get the DataWriterQos.

assert_liveliness (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter) → None

Manually asserts the liveliness of the DataWriter.

close (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter) → None

Close this DataWriter.

create_data (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter) → *rti.connexdds.ParticipantBuiltinTopicData*

Create data of the writer's associated type and initialize it.

property datawriter_cache_status

Get a copy of the cache status for this writer.

property datawriter_protocol_status

Get a copy of the protocol status for this writer.

dispose_instance (**args*, ***kwargs*)

Overloaded function.

1. `dispose_instance(self: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connexdds.InstanceHandle) -> rti.connexdds.ParticipantBuiltinTopicData.DataWriter`

Dispose an instance.

2. `dispose_instance(self: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connexdds.InstanceHandle, timestamp: rti.connexdds.Time) -> rti.connexdds.ParticipantBuiltinTopicData.DataWriter`

Dispose an instance with a timestamp.

3. `dispose_instance(self: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, params: rti.connexdds.WriteParams) -> None`

Dispose an instance with params.

dispose_instance_async (**args*, ***kwargs*)

Overloaded function.

1. `dispose_instance_async(self: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connexdds.InstanceHandle) -> object`

Dispose an instance.

2. `dispose_instance_async(self: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connexdds.InstanceHandle, timestamp: rti.connexdds.Time) -> object`

Dispose an instance with a timestamp.

3. `dispose_instance_async(self: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, key_holder: rti.connexdds.ParticipantBuiltinTopicData) -> object`

Dispose the instance associated with `key_holder`.

4. `dispose_instance_async(self: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, key_holder: rti.connexdds.ParticipantBuiltinTopicData, timestamp: rti.connexdds.Time) -> object`

Dispose the instance associated with `key_holder` using a timestamp

5. `dispose_instance_async(self: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, params: rti.connexdds.WriteParams) -> object`

Dispose an instance with params.

static find_all_by_topic (*publisher*: rti.connexdds.Publisher, *topic_name*: str) → *rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq*

Retrieve all DataWriters for the given topic name in the publisher.

static find_by_name (**args*, ***kwargs*)

Overloaded function.

1. **find_by_name**(*participant*: rti.connexdds.DomainParticipant, *name*: str) -> Optional[*rti.connexdds.ParticipantBuiltinTopicData.DataWriter*]

Find DataWriter in DomainParticipant with the provided name, returning the first found.

2. **find_by_name**(*publisher*: rti.connexdds.Publisher, *name*: str) -> Optional[*rti.connexdds.ParticipantBuiltinTopicData.DataWriter*]

Find DataWriter in Publisher with the DataReader's name, returning the first found.

static find_by_topic (*publisher*: rti.connexdds.Publisher, *name*: str) → Optional[*rti.connexdds.ParticipantBuiltinTopicData.DataWriter*]

Find DataWriter in publisher with a topic name, returning the first found.

flush (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter) → None

Flushes the batch in progress in the context of the calling thread.

is_matched_subscription_active (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, *arg0*: rti.connexdds.InstanceHandle) → bool

A boolean indicating whether or not the matched subscription is active.

is_sample_app_acked (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, *sample_id*: rti.connexdds.SampleIdentity) → bool

Indicates if a sample is considered application-acknowledged.

key_value (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.ParticipantBuiltinTopicData*

Retrieve the instance key that corresponds to an instance handle.

property listener

Get the listener associated with the DataWriter or set the listener.

property liveliness_lost_status

Get a copy of the LivelinessLostStatus.

lookup_instance (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, *key_holder*: rti.connexdds.ParticipantBuiltinTopicData) → *rti.connexdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

matched_subscription_data (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.SubscriptionBuiltinTopicData*

Get the SubscriptionBuiltinTopicData for a subscription matched to this DataWriter.

matched_subscription_datawriter_protocol_status (*args, **kwargs)

Overloaded function.

1. `matched_subscription_datawriter_protocol_status(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.DataWriterProtocolStatus`

Get a copy of the protocol status for this writer per a matched subscription handle.

2. `matched_subscription_datawriter_protocol_status(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, locator: rti.connextdds.Locator) -> rti.connextdds.DataWriterProtocolStatus`

Get a copy of the protocol status for this writer per a matched subscription locator.

matched_subscription_participant_data (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a subscription matched to this DataWriter.

property matched_subscriptions

Get a copy of the list of the currently matched subscription handles.

property matched_subscriptions_locators

The locators used to communicate with matched DataReaders.

property offered_deadline_missed_status

Get a copy of the OfferedDeadlineMissedStatus.

property offered_incompatible_qos_status

Get a copy of the OfferedIncompatibleQosStatus

property publication_matched_status

Get a copy of the PublicationMatchedStatus

property publisher

Get the Publisher that owns this DataWriter.

property qos

The DataWriterQos for this DataWriter. This property's getter returns a deep copy.

register_instance (*args, **kwargs)

Overloaded function.

1. `register_instance(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, key_holder: rti.connextdds.ParticipantBuiltinTopicData) -> rti.connextdds.InstanceHandle`

Informs RTI Connex that the application will be modifying a particular instance.

2. `register_instance(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, key_holder: rti.connextdds.ParticipantBuiltinTopicData, timestamp: rti.connextdds.Time) -> rti.connextdds.InstanceHandle`

Informs RTI Connex that the application will be modifying a particular instance and specified the timestamp.

3. `register_instance(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, key_holder: rti.connextdds.ParticipantBuiltinTopicData, params: rti.connextdds.WriteParams) -> rti.connextdds.InstanceHandle`

Registers instance with parameters.

property reliable_reader_activity_changed_status

Get a copy of the reliable reader activity changed status for this writer.

property reliable_writer_cache_changed_status

Get a copy of the reliable cache status for this writer.

property service_request_accepted_status

Get a copy of the service request accepted status for this writer.

set_listener (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *listener*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener, *event_mask*: rti.connextdds.StatusMask) → None

Set the listener and event mask for the DataWriter.

property topic

Get the Topic object associated with this DataWriter.

property topic_name

Get the topic name associated with this DataWriter.

property type_name

Get the type name for the topic object associated with this DataWriter.

unregister_instance (**args*, ***kwargs*)

Overloaded function.

1. `unregister_instance(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.ParticipantBuiltinTopicData.DataWriter`

Unregister an instance.

2. `unregister_instance(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.ParticipantBuiltinTopicData.DataWriter`

Unregister an instance with timestamp.

3. `unregister_instance(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> None`

Unregister an instance with parameters.

unregister_instance_async (**args*, ***kwargs*)

Overloaded function.

1. `unregister_instance_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> object`

Unregister an instance.

2. `unregister_instance_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Unregister an instance with timestamp.

3. `unregister_instance_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, key_holder: rti.connextdds.ParticipantBuiltinTopicData) -> object`

Unregister the instance associated with `key_holder`.

4. `unregister_instance_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, key_holder: rti.connextdds.ParticipantBuiltinTopicData, timestamp: rti.connextdds.Time) -> object`

Unregister the instance associate with `key_holder` using a timestamp.

5. `unregister_instance_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> object`

Unregister an instance with parameters.

`wait_for_acknowledgments` (*self*:
 rti.connextdds.ParticipantBuiltinTopicData.DataWriter,
 max_wait: rti.connextdds.Duration) → None

Blocks the calling thread until all data written by a reliable `DataWriter` is acknowledged or until the timeout expires.

`wait_for_asynchronous_publishing` (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *max_wait*:
 rti.connextdds.Duration) → None

This operation blocks the calling thread (up to `max_wait`) until all data written by the asynchronous `DataWriter` is sent and acknowledged (if reliable) by all matched `DataReader` entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that `max_wait` elapsed before all the data was sent and/or acknowledged.

In other words, this guarantees that sending to best effort `DataReader` is complete in addition to what `DataWriter.wait_for_acknowledgments()` provides.

If the `DataWriter` does not have `PublishMode` kind set to `PublishModeKind.ASYNCHRONOUS` the operation will complete immediately

`wait_for_asynchronous_publishing_async` (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *max_wait*:
 rti.connextdds.Duration) → object

This function is awaitable until either a timeout of `max_wait` or all data written by the asynchronous `DataWriter` is sent and acknowledged (if reliable) by all matched `DataReader` entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that `max_wait` elapsed before all the data was sent and/or acknowledged. This function works with `asyncio`.

In other words, this guarantees that sending to best effort `DataReader` is complete in addition to what `DataWriter.wait_for_acknowledgments()` provides.

If the `DataWriter` does not have `PublishMode` kind set to `PublishModeKind.ASYNCHRONOUS` the operation will complete immediately

`write` (**args*, ***kwargs*)

Overloaded function.

1. `write(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, samples: rti.connextdds.ParticipantBuiltinTopicDataSeq) -> None`

Write a sequence of samples.

2. `write(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, samples: rti.connextdds.ParticipantBuiltinTopicDataSeq, timestamp: rti.connextdds.Time) -> None`

Write a sequence of samples with a timestamp.

3. `write(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, sample: rti.connextdds.ParticipantBuiltinTopicData) -> None`

Write a sample.

4. `write(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, sample: rti.connextdds.ParticipantBuiltinTopicData, timestamp: rti.connextdds.Time) -> None`

Write a sample with a specified timestamp.

5. `write(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, sample: rti.connextdds.ParticipantBuiltinTopicData, handle: rti.connextdds.InstanceHandle) -> None`

Write a sample with an instance handle.

6. `write(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, sample: rti.connextdds.ParticipantBuiltinTopicData, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> None`

Write a sample with an instance handle and specified timestamp.

7. `write(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, sample: rti.connextdds.ParticipantBuiltinTopicData, params: rti.connextdds.WriteParams) -> None`

Write with advanced parameters.

write_async (*args, **kwargs)

Overloaded function.

1. `write_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, sample: rti.connextdds.ParticipantBuiltinTopicData) -> object`

Write a sample. This method is awaitable and is only for use with asyncio.

2. `write_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, sample: rti.connextdds.ParticipantBuiltinTopicData, timestamp: rti.connextdds.Time) -> object`

Write a sample with a specified timestamp. This methods is awaitable and only for use with asyncio.

3. `write_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, sample: rti.connextdds.ParticipantBuiltinTopicData, handle: rti.connextdds.InstanceHandle) -> object`

Write a sample with an instance handle. This method is awaitable and only for use with asyncio.

4. `write_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, sample: rti.connextdds.ParticipantBuiltinTopicData, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Write a sample with an instance handle and specified timestamp. This method is awaitable and only for use with asyncio.

5. `write_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, samples: rti.connextdds.ParticipantBuiltinTopicDataSeq) -> object`

Write a sequence of samples. This method is awaitable and only for use with asyncio.

6. `write_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, samples: rti.connextdds.ParticipantBuiltinTopicDataSeq, timestamp: rti.connextdds.Time) -> object`

Write a sequence of samples with a timestamp. This method is awaitable and only for use with asyncio.

7. `write_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, samples:`

rti.connextdds.ParticipantBuiltinTopicDataSeq, handles: rti.connextdds.InstanceHandleSeq) -> object

Write a sequence of samples with their instance handles. This method is awaitable and only for use with asyncio.

8. write_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, samples: rti.connextdds.ParticipantBuiltinTopicDataSeq, handles: rti.connextdds.InstanceHandleSeq, timestamp: rti.connextdds.Time) -> object

Write a sequence of samples with their instance handles and a timestamp. This method is awaitable and only for use with asyncio.

9. write_async(self: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, sample: rti.connextdds.ParticipantBuiltinTopicData, params: rti.connextdds.WriteParams) -> object

Write with advanced parameters.

class DataWriterListener

Bases: pybind11_object

__init__ (self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener) → None

__module__ = 'rti.connextdds'

on_application_acknowledgment (self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener, arg0: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg1: rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

on_instance_replaced (self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener, arg0: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg1: rti.connextdds.InstanceHandle) → None

On instance replaced callback.

on_liveliness_lost (self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener, arg0: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg1: rti.connextdds.LivelinessLostStatus) → None

Liveliness lost callback.

on_offered_deadline_missed (self: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener, arg0: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg1: rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

on_offered_incompatible_qos (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

on_publication_matched (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class DataWriterSeq

Bases: pybind11_object

__add__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) → rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq

__bool__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *x*: rti.connexdds.ParticipantBuiltinTopicData.DataWriter) → bool
Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
Overloaded function.
1. **__delitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: int) → None
Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: slice) → None
Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq) → bool

__getitem__ (**args*, ***kwargs*)
Overloaded function.
1. **__getitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *s*: slice) → rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq
Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: int) → rti.connexdds.ParticipantBuiltinTopicData.DataWriter

__hash__ = None

__iadd__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq) → rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq

__imul__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: int) → rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq

__init__ (**args*, ***kwargs*)
Overloaded function.
1. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq) → None
2. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq) → None
Copy constructor
3. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: Iterable) → None

__iter__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq) → Iterator

__len__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: int) → rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq

__ne__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) → bool

__rmul__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: int) → rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: int, *arg1*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter) -> None
2. **__setitem__**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *arg0*: slice, *arg1*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *x*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → None

Add an item to the end of the list

clear (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) → None

Clear the contents

count (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *x*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *L*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *i*: int, *x*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq) -> rti.connextdds.ParticipantBuiltinTopicData.DataWriter

Remove and return the last item

2. **pop**(*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *i*: int) -> rti.connextdds.ParticipantBuiltinTopicData.DataWriter

Remove and return the item at index *i*

remove (*self*: rti.connextdds.ParticipantBuiltinTopicData.DataWriterSeq, *x*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class ITopicDescriptionBases: *IEntity***__init__** (*args, **kwargs)**__module__** = 'rti.connexdds'**property name**

The name of the entity conforming to the ITopicDescription interface.

property participant

The parent DomainParticipant.

property type_name

The name of the associated type.

class LoanedSampleBases: *pybind11_object***__init__** (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.ParticipantBuiltinTopicData.LoanedSample) -> None

Basic constructor

2. **__init__**(self: rti.connexdds.ParticipantBuiltinTopicData.LoanedSample, data: rti.connexdds.ParticipantBuiltinTopicData, info: rti.connexdds.SampleInfo) -> None

Construct LoanedSample with data and info.

__iter__ (self: rti.connexdds.ParticipantBuiltinTopicData.LoanedSample) → object**__module__** = 'rti.connexdds'**property data**

Get the data associated with the sample.

property info

Get the info associated with the sample.

class LoanedSamplesBases: *pybind11_object***__enter__** (self: rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples) → *rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples*

Enter a context for the loaned samples, loan returned on context exit.

__exit__ (self: rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples, arg0: object, arg1: object, arg2: object) → None

Exit the context for the loaned samples, returning the resources.

__getitem__ (self: rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples, arg0: int) → *rti.connexdds.ParticipantBuiltinTopicData.LoanedSample*

Access a LoanedSample object in an array-like syntax

__init__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples) → None
Create an empty LoanedSamples object.

__iter__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples) → Iterator

__len__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples) → int
Get the number of samples in the loan.

__module__ = 'rti.connexdds'

property length

Get the number of samples in the loan.

return_loan (*self*: rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples) → None
Returns the loan to the DataReader.

class NoOpDataReaderListener

Bases: *DataReaderListener*

__init__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener) → None

__module__ = 'rti.connexdds'

on_data_available (*self*: rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataReader) → None
Data available callback.

on_liveliness_changed (*self*: rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connexdds.LivelinessChangedStatus) → None
Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedDeadlineMissedStatus) → None
Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedIncompatibleQosStatus) → None
Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connextdds.SampleLostStatus) → None

Sample lost callback.

on_sample_rejected (*self*: rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connextdds.ParticipantBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataReader, *arg1*: rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class NoOpDataWriterListener

Bases: *DataWriterListener*

__init__ (*self*: rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener) → None

__module__ = 'rti.connextdds'

on_application_acknowledgment (*self*: rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.AcknowledgmentInfo) → None

On application acknowledgment callback

on_instance_replaced (*self*: rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.InstanceHandle) → None

On instance replaced callback.

on_liveliness_lost (*self*: rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.LivelinessLostStatus) → None

Liveliness lost callback.

on_offered_deadline_missed (*self*: rti.connextdds.ParticipantBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connextdds.ParticipantBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

```
on_offered_incompatible_qos (self: rti.connextdds.ParticipantBuiltinTopic-
    Data.NoOpDataWriterListener, arg0:
    rti.connextdds.ParticipantBuiltinTopic-
    Data.DataWriter, arg1:
    rti.connextdds.OfferedIncompatibleQosStatus) →
    None
```

Offered incompatible QoS callback.

```
on_publication_matched (self: rti.connextdds.ParticipantBuiltinTopicData.NoOp-
    DataWriterListener, arg0:
    rti.connextdds.ParticipantBuiltinTopicData.DataWriter, arg1:
    rti.connextdds.PublicationMatchedStatus) → None
```

Publication matched callback.

```
on_reliable_reader_activity_changed (self: rti.connextdds.ParticipantBuiltin-
    TopicData.NoOpDataWriterListener,
    arg0: rti.connextdds.ParticipantBuiltin-
    TopicData.DataWriter, arg1:
    rti.connextdds.ReliableReaderActivity-
    ChangedStatus) →
    None
```

Reliable reader activity changed callback.

```
on_reliable_writer_cache_changed (self: rti.connextdds.ParticipantBuiltinTopic-
    Data.NoOpDataWriterListener, arg0:
    rti.connextdds.ParticipantBuiltinTopic-
    Data.DataWriter, arg1:
    rti.connextdds.ReliableWriterCacheChanged-
    Status) →
    None
```

Reliable writer cache changed callback.

```
on_service_request_accepted (self: rti.connextdds.ParticipantBuiltinTopic-
    Data.NoOpDataWriterListener, arg0:
    rti.connextdds.ParticipantBuiltinTopic-
    Data.DataWriter, arg1:
    rti.connextdds.ServiceRequestAcceptedStatus) →
    None
```

On service request accepted callback.

```
class NoOpTopicListener
```

```
    Bases: TopicListener
```

```
    __init__ (self: rti.connextdds.ParticipantBuiltinTopicData.NoOpTopicListener) → None
```

```
    __module__ = 'rti.connextdds'
```

on_inconsistent_topic (*self*: rti.connexdds.ParticipantBuiltinTopicData.NoOpTopicListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.Topic, *arg1*: rti.connexdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class Sample

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.Sample, *data*: rti.connexdds.ParticipantBuiltinTopicData, *info*: rti.connexdds.SampleInfo) → None

Construct Sample with data and info.

2. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.Sample, *sample*: rti.connexdds.ParticipantBuiltinTopicData.Sample) → None

Copy constructor.

3. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.Sample) → None

Basic constructor

4. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.Sample, *loaned_sample*: rti.connexdds.ParticipantBuiltinTopicData.LoanedSample) → None

Construct a sample with a loaned sample.

__iter__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.Sample) → object

__module__ = 'rti.connexdds'

property data

The data associated with the sample.

property info

Get the info associated with the sample.

class SharedSamples

Bases: pybind11_object

__getitem__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.SharedSamples, *arg0*: int) → rti.connexdds.ParticipantBuiltinTopicData.LoanedSample

Get the sample at the specified index.

__init__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.SharedSamples, *loaned_samples*: rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples) → None

Constructs an instance of SharedSamples, removing ownership of the loan from the Loaned Samples.

__iter__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.SharedSamples) → Iterator

Make a sample iterator

__len__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.SharedSamples) → int

Returns the number of samples.

```
__module__ = 'rti.connexdds'
```

class Topic

Bases: *ITopicDescription, IAnyTopic*

```
__eq__ (self: rti.connexdds.ParticipantBuiltinTopicData.Topic, arg0:
        rti.connexdds.ParticipantBuiltinTopicData.Topic) → bool
```

Test for equality.

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connexdds.ParticipantBuiltinTopicData.Topic, entity: rti.connexdds.IEntity) -> None`

Cast an Entity to a Topic.

2. `__init__(self: rti.connexdds.ParticipantBuiltinTopicData.Topic, topic_description: rti.connexdds.ParticipantBuiltinTopicData.ITopicDescription) -> None`

Cast an ITopicDescription to a Topic.

3. `__init__(self: rti.connexdds.ParticipantBuiltinTopicData.Topic, topic: rti.connexdds.IAnyTopic) -> None`

Create a typed Topic from an AnyTopic.

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.ParticipantBuiltinTopicData.Topic, arg0:
        rti.connexdds.ParticipantBuiltinTopicData.Topic) → bool
```

Test for inequality.

```
static find (participant: rti.connexdds.DomainParticipant, name: str) →
             Optional[rti.connexdds.ParticipantBuiltinTopicData.Topic]
```

Look up a Topic by its name in the DomainParticipant.

property inconsistent_topic_status

Get a copy of the current InconsistentTopicStatus for this Topic.

property listener

The listener.

property qos

Get the TopicQos for this Topic.

This property's getter returns a deep copy.

```
set_listener (self: rti.connexdds.ParticipantBuiltinTopicData.Topic, listener:
               rti.connexdds.ParticipantBuiltinTopicData.TopicListener, event_mask:
               rti.connexdds.StatusMask) → None
```

Set the listener and event mask.

class TopicDescription

Bases: *ITopicDescription*

__eq__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.TopicDescription, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.TopicDescription) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.TopicDescription, *topic*: rti.connexdds.ParticipantBuiltinTopicData.ITopicDescription) -> None

Cast a Topic to a TopicDescription.

2. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.TopicDescription, *entity*: rti.connexdds.IEntity) -> None

Cast a Topic to a TopicDescription.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.TopicDescription, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.TopicDescription) → bool

Test for inequality.

class TopicListener

Bases: pybind11_object

__init__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.TopicListener) → None

__module__ = 'rti.connexdds'

on_inconsistent_topic (*self*: rti.connexdds.ParticipantBuiltinTopicData.TopicListener, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.Topic, *arg1*: rti.connexdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class TopicSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.TopicSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicData.TopicSeq) → rti.connexdds.ParticipantBuiltinTopicData.TopicSeq

__bool__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.TopicSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.TopicSeq, *x*: rti.connexdds.ParticipantBuiltinTopicData.Topic) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicData.TopicSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. `__delitem__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) -> bool`

`__getitem__(*args, **kwargs)`

Overloaded function.

1. `__getitem__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, s: slice) -> rti.connextdds.ParticipantBuiltinTopicData.TopicSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: int) -> rti.connextdds.ParticipantBuiltinTopicData.Topic`

`__hash__ = None`

`__iadd__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) -> rti.connextdds.ParticipantBuiltinTopicData.TopicSeq`

`__imul__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: int) -> rti.connextdds.ParticipantBuiltinTopicData.TopicSeq`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) -> None`
2. `__init__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) -> Iterator`

`__len__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) -> int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: int) -> rti.connextdds.ParticipantBuiltinTopicData.TopicSeq`

`__ne__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) -> bool`

`__rmul__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: int) -> rti.connextdds.ParticipantBuiltinTopicData.TopicSeq`

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: int, arg1: rti.connextdds.ParticipantBuiltinTopicData.Topic) -> None`
2. `__setitem__(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, arg0: slice, arg1: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) -> None`

Assign list elements using a slice object

append (self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, x: rti.connextdds.ParticipantBuiltinTopicData.Topic) → None

Add an item to the end of the list

clear (self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) → None

Clear the contents

count (self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, x: rti.connextdds.ParticipantBuiltinTopicData.Topic) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, L: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, i: int, x: rti.connextdds.ParticipantBuiltinTopicData.Topic) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq) -> rti.connextdds.ParticipantBuiltinTopicData.Topic`

Remove and return the last item

2. `pop(self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, i: int) -> rti.connextdds.ParticipantBuiltinTopicData.Topic`

Remove and return the item at index i

remove (self: rti.connextdds.ParticipantBuiltinTopicData.TopicSeq, x: rti.connextdds.ParticipantBuiltinTopicData.Topic) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class ValidLoanedSamples

Bases: pybind11_object

__enter__ (self: rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples) → rti.connextdds.ParticipantBuiltinTopicData.ValidLoanedSamples

__exit__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ValidLoanedSamples, *arg0*: object, *arg1*: object, *arg2*: object) → None

__init__ (*args, **kwargs)

__iter__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.ValidLoanedSamples) → Iterator

__module__ = 'rti.connexdds'

class WriterContentFilter

Bases: *ContentFilter*

__init__ (*self*: rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter) → None

__module__ = 'rti.connexdds'

writer_attach (*self*: rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter) → Optional[object]

A writer-side filtering API to create some state that can facilitate filtering on the writer side.

writer_compile (*self*: rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object], *property*: rti.connexdds.ExpressionProperty, *expression*: str, *parameters*: rti.connexdds.StringSeq, *type_code*: Optional[rti.connexdds.DynamicType], *type_class_name*: str, *cookie*: rti.connexdds.Cookie) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

writer_detach (*self*: rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object]) → None

A writer-side filtering API to clean up a previously created state using `writer_attach`.

writer_evaluate (*self*: rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object], *sample*: rti.connexdds.ParticipantBuiltinTopicData, *meta_data*: rti.connexdds.FilterSampleInfo) → *rti.connexdds.CookieVector*

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

writer_finalize (*self*: rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object], *cookie*: rti.connexdds.Cookie) → None

A writer-side filtering API to clean up a previously compiled instance of the content filter.

writer_return_loan (*self*: rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object], *cookies*: rti.connexdds.CookieVector) → None

A writer-side filtering API to return the loan on the list of DataReaders returned by `writer_evaluate`.

class WriterContentFilterHelperBases: *WriterContentFilter***__init__** (*self*: rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilterHelper) → None**__module__** = 'rti.connexdds'**add_cookie** (*self*: rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilterHelper, *cookie*: rti.connexdds.Cookie) → None

A helper function which will add a Cookie to the Cookie sequence that is then returned by the `writer_evaluate` function.

writer_evaluate_helper (*self*: rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilterHelper, *writer_filter_data*: *Optional[object]*, *sample*: rti.connexdds.ParticipantBuiltinTopicData, *meta_data*: rti.connexdds.FilterSampleInfo) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching `DataReader`.

__eq__ (*self*: rti.connexdds.ParticipantBuiltinTopicData, *arg0*: rti.connexdds.ParticipantBuiltinTopicData) → bool

Test for equality.

__hash__ = None**__init__** (*self*: rti.connexdds.ParticipantBuiltinTopicData) → None

Create a default `ParticipantBuiltinTopicData`.

__module__ = 'rti.connexdds'**__ne__** (*self*: rti.connexdds.ParticipantBuiltinTopicData, *arg0*: rti.connexdds.ParticipantBuiltinTopicData) → bool

Test for inequality.

property dds_builtin_endpoints

Bitmap of builtin endpoints supported by the participant.

property default_unicast_locators

Get the unicast locators used when individual entities do not specify unicast locators.

property domain_id

Get the domain ID associated with the discovered `DomainParticipant`.

property key

Get the DCPS key to distinguish entries.

property partial_configuration

Indicates whether the `ParticipantBuiltinTopicData` only contains bootstrapping information.

property participant_name

Get the participant name and role name.

property product_version

Get the current version for RTI Connex.

property property

Get the Property policy of the corresponding DomainParticipant.

property reachability_lease_duration

The reachability lease duration.

property rtps_protocol_version

Get the version number of the RTPS wire protocol used.

property rtps_vendor_id

Get the ID of the vendor implementing the RTPS wire protocol.

topic_name = 'DCPSParticipant'

property transport_info

Get the sequence of TransportInfo containing information about each of the installed transports of the discovered DomainParticipant.

property user_data

Get the UserData policy of the corresponding DomainParticipant.

property vendor_builtin_endpoints

Bitmap of vendor-specific builtin endpoints supported by the participant.

class rti.connexdds.ParticipantBuiltinTopicDataSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicDataSeq) → rti.connexdds.ParticipantBuiltinTopicDataSeq

__bool__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *x*: rti.connexdds.ParticipantBuiltinTopicData) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicDataSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *s*: slice) -> rti.connexdds.ParticipantBuiltinTopicDataSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: int) -> rti.connexdds.ParticipantBuiltinTopicData

__hash__ = None

__iadd__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicDataSeq) → *rti.connexdds.ParticipantBuiltinTopicDataSeq*

__imul__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: int) → *rti.connexdds.ParticipantBuiltinTopicDataSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq) -> None
2. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicDataSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq) → Iterator

__len__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: int) → *rti.connexdds.ParticipantBuiltinTopicDataSeq*

__ne__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: rti.connexdds.ParticipantBuiltinTopicDataSeq) → bool

__rmul__ (*self*: rti.connexdds.ParticipantBuiltinTopicDataSeq, *arg0*: int) → *rti.connexdds.ParticipantBuiltinTopicDataSeq*

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, arg0: int, arg1: rti.connextdds.ParticipantBuiltinTopicData) -> None`
2. `__setitem__(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, arg0: slice, arg1: rti.connextdds.ParticipantBuiltinTopicDataSeq) -> None`

Assign list elements using a slice object

append (*self*: rti.connextdds.ParticipantBuiltinTopicDataSeq, *x*: rti.connextdds.ParticipantBuiltinTopicData) → None

Add an item to the end of the list

clear (*self*: rti.connextdds.ParticipantBuiltinTopicDataSeq) → None

Clear the contents

count (*self*: rti.connextdds.ParticipantBuiltinTopicDataSeq, *x*: rti.connextdds.ParticipantBuiltinTopicData) → int

Return the number of times *x* appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, L: rti.connextdds.ParticipantBuiltinTopicDataSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.ParticipantBuiltinTopicDataSeq, *i*: int, *x*: rti.connextdds.ParticipantBuiltinTopicData) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.ParticipantBuiltinTopicDataSeq) -> rti.connextdds.ParticipantBuiltinTopicData`

Remove and return the last item

2. `pop(self: rti.connextdds.ParticipantBuiltinTopicDataSeq, i: int) -> rti.connextdds.ParticipantBuiltinTopicData`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.ParticipantBuiltinTopicDataSeq, *x*: rti.connextdds.ParticipantBuiltinTopicData) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`

Bases: `pybind11_object`

__add__ (*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`, *arg0*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`) → `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`

__bool__ (*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`) → `bool`
Check whether the list is nonempty

__contains__ (*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`, *x*: `Tuple[rti.connexdds.ParticipantBuiltinTopicData, rti.connexdds.Time]`) → `bool`
Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__delitem__**(*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`, *arg0*: `int`) → `None`

Delete the list elements at index *i*

2. **__delitem__**(*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`, *arg0*: `slice`) → `None`

Delete list elements using a slice object

__eq__ (*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`, *arg0*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`) → `bool`

__getitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__getitem__**(*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`, *s*: `slice`) → `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`

Retrieve list elements using a slice object

2. **__getitem__**(*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`, *arg0*: `int`) → `Tuple[rti.connexdds.ParticipantBuiltinTopicData, rti.connexdds.Time]`

__hash__ = `None`

__iadd__ (*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`, *arg0*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`) → `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`

__imul__ (*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`, *arg0*: `int`) → `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: `rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq`) → `None`

2. `__init__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, arg0: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) → Iterator`

`__len__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) → int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, arg0: int) → rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq`

`__ne__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, arg0: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) → bool`

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>`

`__rmul__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, arg0: int) → rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq`

`__setitem__(*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, arg0: int, arg1: Tuple[rti.connextdds.ParticipantBuiltinTopicData, rti.connextdds.Time]) -> None`
2. `__setitem__(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, arg0: slice, arg1: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) -> None`

Assign list elements using a slice object

`append(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, x: Tuple[rti.connextdds.ParticipantBuiltinTopicData, rti.connextdds.Time]) → None`

Add an item to the end of the list

`clear(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) → None`

Clear the contents

`count(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, x: Tuple[rti.connextdds.ParticipantBuiltinTopicData, rti.connextdds.Time]) → int`

Return the number of times `x` appears in the list

`extend(*args, **kwargs)`

Overloaded function.

1. `extend(self: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq, L: rti.connextdds.ParticipantBuiltinTopicDataTimestampedSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq, *i*: int, *x*: Tuple[rti.connexdds.ParticipantBuiltinTopicData, rti.connexdds.Time]) -> None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq) -> Tuple[rti.connexdds.ParticipantBuiltinTopicData, rti.connexdds.Time]`

Remove and return the last item

2. `pop(self: rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq, i: int) -> Tuple[rti.connexdds.ParticipantBuiltinTopicData, rti.connexdds.Time]`

Remove and return the item at index *i*

remove (*self*: rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq, *x*: Tuple[rti.connexdds.ParticipantBuiltinTopicData, rti.connexdds.Time]) -> None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.Partition

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.Partition, *arg0*: rti.connexdds.Partition) -> bool
Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.Partition) -> None`

Creates a policy with the default partition.

2. `__init__(self: rti.connexdds.Partition, partition: str) -> None`

Creates a policy with a single partition with the specified name.

3. `__init__(self: rti.connexdds.Partition, partitions: rti.connexdds.StringSeq) -> None`

Creates a policy with the partitions specified in a sequence.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.Partition, *arg0*: rti.connexdds.Partition) -> bool
Test for inequality.

property name

The partition names specified in a sequence.

```
class rti.connexdds.PersistentJournalKind
```

```
Bases: pybind11_object
```

```
Members:
```

```
DELETE : Deletes the rollback journal at the conclusion of each transaction
```

```
TRUNCATE : Commits transactions by truncating the rollback journal to zero-length instead of deleting it
```

```
PERSIST : Prevents the rollback journal from being deleted at the end of each transaction. Instead, the header of the journal is overwritten with zeros
```

```
MEMORY : Stores the rollback journal in volatile RAM. This saves disk I/O
```

```
WAL : Uses a write-ahead log instead of a rollback journal to implement transactions
```

```
OFF : Completely disables the rollback journal
```

```
DELETE = <PersistentJournalKind.DELETE: 0>
```

```
MEMORY = <PersistentJournalKind.MEMORY: 3>
```

```
OFF = <PersistentJournalKind.OFF: 5>
```

```
PERSIST = <PersistentJournalKind.PERSIST: 2>
```

```
TRUNCATE = <PersistentJournalKind.TRUNCATE: 1>
```

```
WAL = <PersistentJournalKind.WAL: 4>
```

```
__eq__(self: object, other: object) → bool
```

```
__getstate__(self: object) → int
```

```
__hash__(self: object) → int
```

```
__index__(self: rti.connexdds.PersistentJournalKind) → int
```

```
__init__(self: rti.connexdds.PersistentJournalKind, value: int) → None
```

```
__int__(self: rti.connexdds.PersistentJournalKind) → int
```

```
__members__ = {'DELETE': <PersistentJournalKind.DELETE: 0>,
'MEMORY': <PersistentJournalKind.MEMORY: 3>, 'OFF':
<PersistentJournalKind.OFF: 5>, 'PERSIST':
<PersistentJournalKind.PERSIST: 2>, 'TRUNCATE':
<PersistentJournalKind.TRUNCATE: 1>, 'WAL':
<PersistentJournalKind.WAL: 4>}
```

```
__module__ = 'rti.connexdds'
```

`__ne__` (*self*: object, *other*: object) → bool

`__repr__` (*self*: object) → str

`__setstate__` (*self*: rti.connexdds.PersistentJournalKind, *state*: int) → None

`__str__` ()

name(*self*: handle) -> str

property name

property value

class rti.connexdds.PersistentStorageSettings

Bases: pybind11_object

`__eq__` (*self*: rti.connexdds.PersistentStorageSettings, *arg0*:
rti.connexdds.PersistentStorageSettings) → bool

Test for equality.

`__hash__` = None

`__init__` (*self*: rti.connexdds.PersistentStorageSettings) → None

Create a PersistentStorageSettings with default settings.

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.PersistentStorageSettings, *arg0*:
rti.connexdds.PersistentStorageSettings) → bool

Test for inequality.

property enable

Enables durable writer history in a DataWriter and durable reader state in a DataReader.

property file_name

The file name where the durable writer history or durable reader state will be stored.

property journal_kind

Sets the journal mode of the persistent storage.

property reader_checkpoint_frequency

Controls how often the reader state is stored into the database.

property restore

Indicates if the persisted writer history or reader state must be restored.

property synchronization_kind

Sets the journal mode of the persistent storage.

property trace_file_name

The file name where to store the SQL statements executed when loading and storing the durable writer history or durable reader state.

property vacuum

Sets the auto-vacuum status of the storage.

property writer_instance_cache_allocation

Configures the resource limits associated with the instance durable writer history cache.

property writer_memory_state

Determines how much state will be kept in memory by the durable writer history in order to avoid accessing the persistent storage in disk.

property writer_sample_cache_allocation

Configures the resource limits associated with the sample durable writer history cache.

class rti.connexdds.PersistentSynchronizationKind

Bases: pybind11_object

Members:

NORMAL : Data (e.g., new sample) is written to disk at critical moments

FULL : Data (e.g., new sample) is written to physical disk immediately

OFF : No synchronization is enforced

FULL = <PersistentSynchronizationKind.FULL: 1>

NORMAL = <PersistentSynchronizationKind.NORMAL: 0>

OFF = <PersistentSynchronizationKind.OFF: 2>

__eq__ (*self: object, other: object*) → bool

__getstate__ (*self: object*) → int

__hash__ (*self: object*) → int

__index__ (*self: rti.connexdds.PersistentSynchronizationKind*) → int

__init__ (*self: rti.connexdds.PersistentSynchronizationKind, value: int*) → None

__int__ (*self: rti.connexdds.PersistentSynchronizationKind*) → int

__members__ = {'FULL': <PersistentSynchronizationKind.FULL: 1>, 'NORMAL': <PersistentSynchronizationKind.NORMAL: 0>, 'OFF': <PersistentSynchronizationKind.OFF: 2>}

__module__ = 'rti.connexdds'

__ne__ (*self: object, other: object*) → bool

__repr__ (*self: object*) → str

__setstate__ (*self: rti.connexdds.PersistentSynchronizationKind, state: int*) → None

```
__str__()
    name(self: handle) -> str
```

property name

property value

exception `rti.connextdds.PreconditionNotMetError`

Bases: *Exception*

```
__module__ = 'rti.connextdds'
```

class `rti.connextdds.Presentation`

Bases: `pybind11_object`

```
__eq__(self: rti.connextdds.Presentation, arg0: rti.connextdds.Presentation) → bool
    Test for equality.
```

```
__hash__ = None
```

```
__init__(*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connextdds.Presentation) -> None`

Create an instance with the default value.

2. `__init__(self: rti.connextdds.Presentation, access_scope: rti.connextdds.PresentationAccessScopeKind, coherent_access: bool, ordered_access: bool) -> None`

Creates an instance with the specified access scope and coherent and ordered access.

```
__module__ = 'rti.connextdds'
```

```
__ne__(self: rti.connextdds.Presentation, arg0: rti.connextdds.Presentation) → bool
    Test for inequality.
```

property access_scope

Determines the largest scope spanning the entities for which the order and coherency of changes can be preserved.

property coherent_access

Controls whether coherent access is supported within the `access_scope`.

property drop_incomplete_coherent_set

Indicates whether or not a `DataReader` should drop samples from an incomplete coherent set

static `group_access_scope` (*coherent: bool, ordered: bool*) → `rti.connextdds.Presentation`

Creates a `Presentation` instance with group access scope.

static `instance_access_scope` (*coherent: bool, ordered: bool*) → `rti.connextdds.Presentation`

Creates a `Presentation` instance with instance access scope.

property ordered_access

Controls whether ordered access is supported within the access_scope.

static topic_access_scope (*coherent: bool, ordered: bool*) → *rti.connexdds.Presentation*

Creates a Presentation instance with topic access scope.

class *rti.connexdds.PresentationAccessScopeKind*

Bases: *pybind11_object*

GROUP = <*PresentationAccessScopeKind.GROUP: 2*>

HIGHEST_OFFERED = <*PresentationAccessScopeKind.HIGHEST_OFFERED: 3*>

INSTANCE = <*PresentationAccessScopeKind.INSTANCE: 0*>

class *PresentationAccessScopeKind*

Bases: *pybind11_object*

Members:

INSTANCE : [default] Scope spans only a single instance.

Indicates that changes to one instance need not be coherent nor ordered with respect to changes to any other instance. In other words, order and coherent changes apply to each instance separately.

TOPIC : Scope spans to all instances within the same DataWriter (or DataReader), but not across instances in different DataWriter (or DataReader).

GROUP : Scope spans to all instances belonging to DataWriter (or DataReader) entities within the same Publisher (or Subscriber).

HIGHEST_OFFERED : This value only applies to a Subscriber. The Subscriber will use the access scope specified by each remote Publisher.

GROUP = <*PresentationAccessScopeKind.GROUP: 2*>

HIGHEST_OFFERED = <*PresentationAccessScopeKind.HIGHEST_OFFERED: 3*>

INSTANCE = <*PresentationAccessScopeKind.INSTANCE: 0*>

TOPIC = <*PresentationAccessScopeKind.TOPIC: 1*>

__eq__ (*self: object, other: object*) → bool

__getstate__ (*self: object*) → int

__hash__ (*self: object*) → int

__index__ (*self: rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind*) → int

```

__init__(self: rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind,
         value: int) → None

__int__(self: rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind)
→ int

__members__ = {'GROUP': <PresentationAccessScopeKind.GROUP: 2>,
               'HIGHEST_OFFERED':
<PresentationAccessScopeKind.HIGHEST_OFFERED: 3>, 'INSTANCE':
<PresentationAccessScopeKind.INSTANCE: 0>, 'TOPIC':
<PresentationAccessScopeKind.TOPIC: 1>}

__module__ = 'rti.connextdds'

__ne__(self: object, other: object) → bool

__repr__(self: object) → str

__setstate__(self:
             rti.connextdds.PresentationAccessScopeKind.PresentationAccessScopeKind,
             state: int) → None

```

```

__str__()
name(self: handle) -> str

```

property name

property value

TOPIC = <PresentationAccessScopeKind.TOPIC: 1>

```

__eq__(self: rti.connextdds.PresentationAccessScopeKind, arg0:
      rti.connextdds.PresentationAccessScopeKind) → bool

```

Apply operator to underlying enumerated values.

```

__ge__(self: rti.connextdds.PresentationAccessScopeKind, arg0:
      rti.connextdds.PresentationAccessScopeKind) → bool

```

Apply operator to underlying enumerated values.

```

__gt__(self: rti.connextdds.PresentationAccessScopeKind, arg0:
      rti.connextdds.PresentationAccessScopeKind) → bool

```

Apply operator to underlying enumerated values.

__hash__ = None

```

__init__(*args, **kwargs)

```

Overloaded function.

1. `__init__(self: rti.connextdds.PresentationAccessScopeKind) -> None`

Initializes enum to 0.

2. `__init__(self: rti.connexdds.PresentationAccessScopeKind, arg0: rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind) -> None`

Copy constructor.

`__int__(self: rti.connexdds.PresentationAccessScopeKind) -> rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind`

Int conversion.

`__le__(self: rti.connexdds.PresentationAccessScopeKind, arg0: rti.connexdds.PresentationAccessScopeKind) -> bool`

Apply operator to underlying enumerated values.

`__lt__(self: rti.connexdds.PresentationAccessScopeKind, arg0: rti.connexdds.PresentationAccessScopeKind) -> bool`

Apply operator to underlying enumerated values.

`__module__ = 'rti.connexdds'`

`__ne__(self: rti.connexdds.PresentationAccessScopeKind, arg0: rti.connexdds.PresentationAccessScopeKind) -> bool`

Apply operator to underlying enumerated values.

`__str__(self: rti.connexdds.PresentationAccessScopeKind) -> str`
String conversion.

property underlying

Retrieves the actual enumerated value.

class `rti.connexdds.PrintFormat`

Bases: `pybind11_object`

DEBUG = `<PrintFormat.DEBUG: 13935>`

DEFAULT = `<PrintFormat.DEFAULT: 13330>`

MAXIMAL = `<PrintFormat.MAXIMAL: 16255>`

MINIMAL = `<PrintFormat.MINIMAL: 17>`

class `PrintFormat`

Bases: `pybind11_object`

Members:

DEFAULT : Print message, method name, and activity context (default).

TIMESTAMPED : Print message, method name, activity context, and timestamp.

VERBOSE : Print message with all available context information (includes thread identifier, activity context).

VERBOSE_TIMESTAMPED : Print message with all available context information, and timestamp.

DEBUG : Print a set of field that may be useful for internal debug.

MINIMAL : Print only message number and method name.

MAXIMAL : Print all available fields.

DEBUG = <PrintFormat.DEBUG: 13935>

DEFAULT = <PrintFormat.DEFAULT: 13330>

MAXIMAL = <PrintFormat.MAXIMAL: 16255>

MINIMAL = <PrintFormat.MINIMAL: 17>

TIMESTAMPED = <PrintFormat.TIMESTAMPED: 13586>

VERBOSE = <PrintFormat.VERBOSE: 15894>

VERBOSE_TIMESTAMPED = <PrintFormat.VERBOSE_TIMESTAMPED: 16150>

`__eq__` (*self: object, other: object*) → bool

`__getstate__` (*self: object*) → int

`__hash__` (*self: object*) → int

`__index__` (*self: rti.connexdds.PrintFormat.PrintFormat*) → int

`__init__` (*self: rti.connexdds.PrintFormat.PrintFormat, value: int*) → None

`__int__` (*self: rti.connexdds.PrintFormat.PrintFormat*) → int

`__members__` = {'DEBUG': <PrintFormat.DEBUG: 13935>, 'DEFAULT': <PrintFormat.DEFAULT: 13330>, 'MAXIMAL': <PrintFormat.MAXIMAL: 16255>, 'MINIMAL': <PrintFormat.MINIMAL: 17>, 'TIMESTAMPED': <PrintFormat.TIMESTAMPED: 13586>, 'VERBOSE': <PrintFormat.VERBOSE: 15894>, 'VERBOSE_TIMESTAMPED': <PrintFormat.VERBOSE_TIMESTAMPED: 16150>}

`__module__` = 'rti.connexdds'

`__ne__` (*self: object, other: object*) → bool

`__repr__` (*self: object*) → str

`__setstate__` (*self: rti.connexdds.PrintFormat.PrintFormat, state: int*) → None

`__str__` ()

name(self: handle) -> str

property name

property value

```

TIMESTAMPED = <PrintFormat.TIMESTAMPED: 13586>
VERBOSE = <PrintFormat.VERBOSE: 15894>
VERBOSE_TIMESTAMPED = <PrintFormat.VERBOSE_TIMESTAMPED: 16150>

__eq__ (self: rti.connexdds.PrintFormat, arg0: rti.connexdds.PrintFormat) → bool
    Apply operator to underlying enumerated values.

__ge__ (self: rti.connexdds.PrintFormat, arg0: rti.connexdds.PrintFormat) → bool
    Apply operator to underlying enumerated values.

__gt__ (self: rti.connexdds.PrintFormat, arg0: rti.connexdds.PrintFormat) → bool
    Apply operator to underlying enumerated values.

__hash__ = None

__init__ (*args, **kwargs)
    Overloaded function.
    1. __init__(self: rti.connexdds.PrintFormat) -> None
        Initializes enum to 0.
    2. __init__(self: rti.connexdds.PrintFormat, arg0: rti.connexdds.PrintFormat.PrintFormat) ->
        None
    Copy constructor.

__int__ (self: rti.connexdds.PrintFormat) → rti.connexdds.PrintFormat.PrintFormat
    Int conversion.

__le__ (self: rti.connexdds.PrintFormat, arg0: rti.connexdds.PrintFormat) → bool
    Apply operator to underlying enumerated values.

__lt__ (self: rti.connexdds.PrintFormat, arg0: rti.connexdds.PrintFormat) → bool
    Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.PrintFormat, arg0: rti.connexdds.PrintFormat) → bool
    Apply operator to underlying enumerated values.

__str__ (self: rti.connexdds.PrintFormat) → str
    String conversion.

property underlying
    Retrieves the actual enumerated value.

class rti.connexdds.PrintFormatKind
    Bases: pybind11_object

    DEFAULT = <PrintFormatKind.DEFAULT: 0>

```

JSON = <PrintFormatKind.JSON: 2>

class PrintFormatKind

Bases: pybind11_object

Members:

DEFAULT : Use a default format specific to RTI Connext to represent the data when converting a data sample to a string.

XML : Use an XML format to represent the data when converting a data sample to a string.

JSON : Use an JSON format to represent the data when converting a data sample to a string.

DEFAULT = <PrintFormatKind.DEFAULT: 0>

JSON = <PrintFormatKind.JSON: 2>

XML = <PrintFormatKind.XML: 1>

__eq__ (*self: object, other: object*) → bool

__getstate__ (*self: object*) → int

__hash__ (*self: object*) → int

__index__ (*self: rti.connextdds.PrintFormatKind.PrintFormatKind*) → int

__init__ (*self: rti.connextdds.PrintFormatKind.PrintFormatKind, value: int*) → None

__int__ (*self: rti.connextdds.PrintFormatKind.PrintFormatKind*) → int

__members__ = {'DEFAULT': <PrintFormatKind.DEFAULT: 0>, 'JSON': <PrintFormatKind.JSON: 2>, 'XML': <PrintFormatKind.XML: 1>}

__module__ = 'rti.connextdds'

__ne__ (*self: object, other: object*) → bool

__repr__ (*self: object*) → str

__setstate__ (*self: rti.connextdds.PrintFormatKind.PrintFormatKind, state: int*) → None

__str__ ()

name(self: handle) -> str

property name

property value

XML = <PrintFormatKind.XML: 1>

__eq__ (*self: rti.connextdds.PrintFormatKind, arg0: rti.connextdds.PrintFormatKind*) → bool

Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.PrintFormatKind, *arg0*: rti.connexdds.PrintFormatKind) → bool
Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.PrintFormatKind, *arg0*: rti.connexdds.PrintFormatKind) → bool
Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: rti.connexdds.PrintFormatKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.PrintFormatKind, *arg0*: rti.connexdds.PrintFormatKind.PrintFormatKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.PrintFormatKind) → *rti.connexdds.PrintFormatKind.PrintFormatKind*
Int conversion.

__le__ (*self*: rti.connexdds.PrintFormatKind, *arg0*: rti.connexdds.PrintFormatKind) → bool
Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.PrintFormatKind, *arg0*: rti.connexdds.PrintFormatKind) → bool
Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.PrintFormatKind, *arg0*: rti.connexdds.PrintFormatKind) → bool
Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.PrintFormatKind) → str
String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.PrintFormatProperty

Bases: pybind11_object

__init__ (*self*: rti.connexdds.PrintFormatProperty, *kind*: rti.connexdds.PrintFormatKind = *PrintFormatKind.DEFAULT*, *is_pretty_print*: bool = True, *is_enum_as_int*: bool = False, *is_include_root_elements*: bool = True) → None

Initialize PrintFormatProperty object.

__module__ = 'rti.connexdds'

default = <rti.connexdds.PrintFormatProperty object>

property enum_as_int

Get/set the value of enum_as_int.

property include_root_elements

Get/set the value of include_root_elements.

json = <rti.connextdds.PrintFormatProperty object>

property kind

Get/set the kind value.

property pretty_print

Get/set the value of pretty_print.

xml = <rti.connextdds.PrintFormatProperty object>

class rti.connextdds.ProductVersion

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.ProductVersion, *arg0*: rti.connextdds.ProductVersion) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.ProductVersion) → None

Creates the unknown product version.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.ProductVersion, *arg0*: rti.connextdds.ProductVersion) → bool

Test for inequality.

__str__ (*self*: rti.connextdds.ProductVersion) → str

current = <rti.connextdds.ProductVersion object>

property major_version

The major product version.

property minor_version

The minor product version.

property native_build_id

A string with the build ID of the native Connext libraries.

property release_version

The release letter of the product version.

property revision_version

The revision number of product.

unknown = <rti.connextdds.ProductVersion object>

class `rti.connexdds.Property`

Bases: `pybind11_object`

__contains__ (*self*: `rti.connexdds.Property`, *arg0*: `str`) → `bool`

__eq__ (*self*: `rti.connexdds.Property`, *arg0*: `rti.connexdds.Property`) → `bool`

Test for equality.

__getitem__ (*self*: `rti.connexdds.Property`, *arg0*: `str`) → `str`

__hash__ = `None`

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: `rti.connexdds.Property`) -> `None`

Creates an empty Property container.

2. **__init__**(*self*: `rti.connexdds.Property`, *entries*: `dict`, *propagate*: `bool` = `False`) -> `None`

Creates a Property container with entries from a dictionary of entries

3. **__init__**(*self*: `rti.connexdds.Property`, *entries*: `rti.connexdds.StringPairSeq`, *propagate*: `bool` = `False`) -> `None`

Creates a Property container with entries from a sequence of entries

__len__ (*self*: `rti.connexdds.Property`) → `int`

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.Property`, *arg0*: `rti.connexdds.Property`) → `bool`

Test for inequality.

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: `rti.connexdds.Property`, *arg0*: `str`, *arg1*: `str`) -> `rti.connexdds.Property`

2. **__setitem__**(*self*: `rti.connexdds.Property`, *arg0*: `str`, *arg1*: `Tuple[str, bool]`) -> `rti.connexdds.Property`

exists (*self*: `rti.connexdds.Property`, *key*: `str`) → `bool`

Returns true if a property exists.

get (*self*: `rti.connexdds.Property`, *key*: `str`) → `str`

Returns the value of a property identified by a key if it exists.

get_all (*self*: `rti.connexdds.Property`) → `rti.connexdds.StringMap`

Retrieves a copy of all the entries in a map.

propagate (*self*: `rti.connexdds.Property`, *key*: `str`) → `bool`

Returns whether the 'propagate' attribute for a property is set or not.

remove (*self*: rti.connexdds.Property, *key*: str) → bool

Removes the property identified by a key.

set (**args*, ***kwargs*)

Overloaded function.

1. set(*self*: rti.connexdds.Property, *entry*: Tuple[str, str], *propagate*: bool = False) -> rti.connexdds.Property

Adds or assigns a property from a pair of strings.

2. set(*self*: rti.connexdds.Property, *key*: str, *value*: str, *propagate*: bool = False) -> rti.connexdds.Property

Adds or assigns a property from a key and a value.

3. set(*self*: rti.connexdds.Property, *entries*: rti.connexdds.StringPairSeq, *propagate*: bool = False) -> rti.connexdds.Property

Adds a range of properties.

4. set(*self*: rti.connexdds.Property, *entries*: dict, *propagate*: bool = False) -> None

Adds a range of properties.

size (*self*: rti.connexdds.Property) → int

Returns the number of properties.

try_get (*self*: rti.connexdds.Property, *key*: str) → Optional[str]

Returns the value of a property identified by a key if it exists.

class rti.connexdds.ProtocolVersion

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.ProtocolVersion, *arg0*: rti.connexdds.ProtocolVersion) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. __init__(*self*: rti.connexdds.ProtocolVersion) -> None

Creates an invalid protocol version.

2. __init__(*self*: rti.connexdds.ProtocolVersion, *major*: int, *minor*: int) -> None

Creates a version with the given major and minor values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ProtocolVersion, *arg0*: rti.connexdds.ProtocolVersion) → bool

Test for inequality.

current = <rti.connexdds.ProtocolVersion object>

property major_version

The major version number.

property minor_version

The minor version number.

class `rti.connexdds.PublicationBuiltinTopicData`

Bases: `pybind11_object`

class `ContentFilter`

Bases: `ContentFilterBase`

__init__ (*self*: `rti.connexdds.PublicationBuiltinTopicData.ContentFilter`) → None

__module__ = `'rti.connexdds'`

compile (*self*: `rti.connexdds.PublicationBuiltinTopicData.ContentFilter`, *expression*: `str`, *parameters*: `rti.connexdds.StringSeq`, *type_code*: `Optional[rti.connexdds.DynamicType]`, *type_class_name*: `str`, *old_compile_data*: `Optional[object]`) → `Optional[object]`

Compile an instance of the content filter according to the filter expression and parameters of the given data type.

evaluate (*self*: `rti.connexdds.PublicationBuiltinTopicData.ContentFilter`, *compile_data*: `Optional[object]`, *sample*: `rti.connexdds.PublicationBuiltinTopicData`, *meta_data*: `rti.connexdds.FilterSampleInfo`) → `bool`

Evaluate whether the sample is passing the filter or not according to the sample content.

finalize (*self*: `rti.connexdds.PublicationBuiltinTopicData.ContentFilter`, *compile_data*: `Optional[object]`) → None

A previously compiled instance of the content filter is no longer in use and resources can now be cleaned up.

class `ContentFilteredTopic`

Bases: `ITopicDescription`, `IAnyTopic`

__eq__ (*self*: `rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic`, *arg0*: `rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic`) → `bool`

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: `rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic`, *topic*: `rti.connexdds.PublicationBuiltinTopicData.Topic`, *name*: `str`, *contentfilter*: `rti.connexdds.Filter`) → None

Create a `ContentFilteredTopic` with a name and Filter.

2. **__init__**(*self*: `rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic`, *topic_description*: `rti.connexdds.PublicationBuiltinTopicData.ITopicDescription`) → None

Cast a TopicDescription to a ContentFilteredTopic.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic, *arg0*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic) → bool

Test for inequality.

append_to_expression_parameter (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic, *index*: int, *extension*: str) → None

Append the extension to the end of parameter at the provided index, separated by a comma.

property filter_expression

Get the filter expression

property filter_parameters

Get/set the filter parameters.

static find (*participant*: rti.connexdds.DomainParticipant, *name*: str) → Optional[rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

remove_from_expression_parameter (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic, *index*: int, *remove_term*: str) → None

Removes the specified term from the parameter at the provided index.

set_filter (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic, *arg0*: rti.connexdds.Filter) → None

Set the filter.

property topic

Get the underlying Topic.

class ContentFilteredTopicSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq

__bool__ (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic) → bool

Return true the container contains x

`__delitem__` (*args, **kwargs)

Overloaded function.

1. `__delitem__(self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> None`

Delete the list elements at index `i`

2. `__delitem__(self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → bool

`__getitem__` (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, s: slice) -> rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic`

`__hash__` = None

`__iadd__` (self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → *rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq*

`__imul__` (self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) → *rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq*

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) -> None`

2. `__init__(self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: Iterable) -> None`

`__iter__` (self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → Iterator

`__len__` (self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → int

`__module__` = 'rti.connexdds'

`__mul__` (self: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) → *rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq*

__ne__ (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → bool

__rmul__ (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) → rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int, *arg1*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic) -> None
2. **__setitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: slice, *arg1*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) → None

Clear the contents

count (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *L*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *i*: int, *x*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq) -> rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic

Remove and return the last item

2. **pop**(*self*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *i*: int) -> rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic

Remove and return the item at index *i*

remove (*self*: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class DataReader

Bases: *IDataReader*

class Selector

Bases: *pybind11_object*

__init__ (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector, *datareader*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → None

Create a Selector for a DataReader to read/take based on selected conditions

__module__ = 'rti.connextdds'

condition (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector, *condition*: rti.connextdds.IReadCondition) → *rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector*

Select samples based on a ReadCondition.

content (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector, *query*: rti.connextdds.Query) → *rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector*

Select samples based on a Query.

instance (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector*

Select a specific instance to read/take.

max_samples (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector, *max*: *int*) → *rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector*

Limit the number of samples read/taken by the Select.

next_instance (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector, *previous*: rti.connextdds.InstanceHandle) → *rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector*

Select the instance after the specified instance to read/take.

read (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector) → list

Read copies of available samples (data and info) based on the Selector settings.

read_data (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector) → list

Read copies of available valid data based on the Selector settings.

read_loaned (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader.Selector) → *rti.connextdds.PublicationBuiltinTopicData.LoanedSamples*

Take available samples (data and info) based on the Selector settings and return them in a loaned container.

state (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector, *state*: rti.connexdds.DataState) → *rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector*

Select samples with a specified data state.

take (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector) → list
Take copies of available samples (data and info) based on the Selector settings.

take_data (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector) → list

Take copies of available valid data based on the Selector settings.

take_loaned (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector) → *rti.connexdds.PublicationBuiltinTopicData.LoanedSamples*

Read available samples (data and info) based on the Selector settings and return them in a loaned container.

__enter__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader) → *rti.connexdds.PublicationBuiltinTopicData.DataReader*

Enter a context for this DataReader, to be cleaned up on exiting context

__eq__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader) → bool

Test for equality.

__exit__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg0*: object, *arg1*: object, *arg2*: object) → None

Exit the context for this DataReader, cleaning up resources.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *topic*: rti.connexdds.PublicationBuiltinTopicData.Topic) -> None

Create a DataReader in the implicit subscriber with default QoS.

2. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *topic*: rti.connexdds.PublicationBuiltinTopicData.Topic, *qos*: rti.connexdds.DataReaderQos, *listener*: rti.connexdds.PublicationBuiltinTopicData.DataReaderListener = None, *mask*: rti.connexdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader in the implicit subscriber with specific QoS and a listener.

3. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *cft*: rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic) -> None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with default QoS.

4. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, cft: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.PublicationBuiltinTopicData.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with specific QoS.

5. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, subscriber: rti.connextdds.Subscriber, topic: rti.connextdds.PublicationBuiltinTopicData.Topic) -> None`

Create a DataReader.

6. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, subscriber: rti.connextdds.Subscriber, topic: rti.connextdds.PublicationBuiltinTopicData.Topic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.PublicationBuiltinTopicData.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Create a DataReader in a subscriber with specific QoS and a listener.

7. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, subscriber: rti.connextdds.Subscriber, cft: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic) -> None`

Create a DataReader with a ContentFilteredTopic in a subscriber with default QoS.

8. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, subscriber: rti.connextdds.Subscriber, cft: rti.connextdds.PublicationBuiltinTopicData.ContentFilteredTopic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.PublicationBuiltinTopicData.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Create a DataReader with a ContentFilteredTopic in a subscriber with specific QoS.

9. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, reader: rti.connextdds.IAnyDataReader) -> None`

Get a typed DataReader from an AnyDataReader.

10. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, entity: rti.connextdds.IEntity) -> None`

Get a typed DataReader from an Entity.

`__lshift__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, arg0: rti.connextdds.DataReaderQos) -> rti.connextdds.PublicationBuiltinTopicData.DataReader`

Set the DataReaderQos for this DataReader.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, arg0: rti.connextdds.PublicationBuiltinTopicData.DataReader) -> bool`

Test for inequality.

`__rshift__(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, arg0: rti.connextdds.DataReaderQos) -> rti.connextdds.PublicationBuiltinTopicData.DataReader`

Get the DataReaderQos from this DataReader

acknowledge_all (*args, **kwargs)

Overloaded function.

1. `acknowledge_all(self: rti.connextdds.PublicationBuiltinTopicData.DataReader) -> None`
Acknowledge all previously accessed samples.
2. `acknowledge_all(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, arg0: rti.connextdds.AckResponseData) -> None`
Acknowledge all previously accessed samples.

acknowledge_sample (*args, **kwargs)

Overloaded function.

1. `acknowledge_sample(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, sample_info: rti.connextdds.SampleInfo) -> None`
Acknowledge a single sample.
2. `acknowledge_sample(self: rti.connextdds.PublicationBuiltinTopicData.DataReader, sample_info: rti.connextdds.SampleInfo, ack_response_data: rti.connextdds.AckResponseData) -> None`
Acknowledge a single sample with ack response data.

close (self: rti.connextdds.PublicationBuiltinTopicData.DataReader) → None

Close this DataReader.

property datareader_cache_status

Get the DataReaderCacheStatus for the DataReader.

property datareader_protocol_status

Get the DataReaderProtocolStatus for the DataReader.

property default_filter_state

Returns the filter state for the read/take operations.

static find_all_by_topic (subscriber: rti.connextdds.Subscriber, topic_name: str) → rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq

Retrieve all DataReaders for the given topic name in the subscriber.

static find_by_name (*args, **kwargs)

Overloaded function.

1. `find_by_name(participant: rti.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.PublicationBuiltinTopicData.DataReader]`
Find DataReader in DomainParticipant with the DataReader's name, returning the first found.
2. `find_by_name(subscriber: rti.connextdds.Subscriber, name: str) -> Optional[rti.connextdds.PublicationBuiltinTopicData.DataReader]`
Find DataReader in Subscriber with the DataReader's name, returning the first found.

static find_by_topic (subscriber: rti.connextdds.Subscriber, name: str) → Optional[rti.connextdds.PublicationBuiltinTopicData.DataReader]

Find DataReader in Subscriber with a topic name, returning the first found.

is_matched_publication_alive (self: rti.connextdds.PublicationBuiltinTopicData.DataReader, arg0: rti.connextdds.InstanceHandle) → bool

Check if a matched publication is alive.

key_value (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.PublicationBuiltinTopicData*

Retrieve the instance key that corresponds to an instance handle.

property listener

Gets or sets the listener with StatusMask.ALL

property liveliness_changed_status

Get the LivelinessChangedStatus for this DataReader.

lookup_instance (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *key_holder*: rti.connexdds.PublicationBuiltinTopicData) → *rti.connexdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

matched_publication_data (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.PublicationBuiltinTopicData*

Get the PublicationBuiltinTopicData for a publication matched to this DataReader.

matched_publication_datareader_protocol_status (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *publication_handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.DataReaderProtocolStatus*

Get the DataReaderProtocolStatus for the DataReader based on the matched publication handle.

matched_publication_participant_data (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a publication matched to this DataReader.

property matched_publications

Get a copy of the list of the currently matched publication handles.

property qos

The DataReaderQos for this DataReader.

This property's getter returns a deep copy.

- read** (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → list
Read copies of all available samples (data and info).
- read_data** (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → list
Read copies of all available valid data.
- read_loaned** (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → *rti.connextdds.PublicationBuiltinTopicData.LoanedSamples*
Read all available samples (data and info) and return them in a loaned container.
- property requested_deadline_missed_status**
Get the RequestedDeadlineMissed status for the DataReader
- property requested_incompatible_qos_status**
Get the RequestedIncompatibleQosStatus for the DataReader.
- property sample_lost_status**
Get the SampleLostStatus for the DataReader.
- property sample_rejected_status**
Get the SampleRejectedStatus for the DataReader.
- select** (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → *dds::sub::DataReader<dds::topic::TPublicationBuiltinTopicData<rti::topic::PublicationBuiltinTopicDataImpl>, rti::sub::DataReaderImpl>::Selector*
Get a Selector to perform complex data selections, such as per-instance selection, content, and status filtering.
- set_listener** (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader, *listener*: *rti.connextdds.PublicationBuiltinTopicData.DataReaderListener*, *event_mask*: *rti.connextdds.StatusMask*) → None
Set the listener and associated event mask.
- property subscriber**
Returns the parent Subscriber of the DataReader.
- property subscription_matched_status**
Get the SubscriptionMatchedStatus for the DataReader.
- take** (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → list
Take copies of all available samples (data and info).
- take_data** (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → list
Take copies of all available valid data.
- take_loaned** (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → *rti.connextdds.PublicationBuiltinTopicData.LoanedSamples*
Take all available samples (data and info) and return them in a loaned container.

property topic_description

Returns the TopicDescription associated with the DataReader.

property topic_name

Get the topic name associated with this DataReader.

property type_name

Get the type name associated with this DataReader.

wait_for_historical_data (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader, *max_wait*: rti.connextdds.Duration) → None

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind.

wait_for_historical_data_async (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReader, *max_wait*: rti.connextdds.Duration) → object

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind. This call is awaitable and only for use with asyncio.

class DataReaderListener

Bases: pybind11_object

__init__ (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderListener) → None

__module__ = 'rti.connextdds'

on_data_available (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderListener, *arg0*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → None

Data available callback.

on_liveliness_changed (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderListener, *arg0*: rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderListener, *arg0*: rti.connextdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SampleLostStatus) → None

Sample lost callback.

on_sample_rejected (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class DataReaderSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq) → rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq

__bool__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.DataReader) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq, *arg0*: int) → None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq, *arg0*: slice) → None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq) → bool

__getitem__ (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, s: slice)`
-> `rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: int)` -> `rti.connextdds.PublicationBuiltinTopicData.DataReader`

__hash__ = None

__iadd__ (self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq) → rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq

__imul__ (self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: int) → rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq)` -> None
2. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq)` -> None

Copy constructor

3. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: Iterable)` -> None

__iter__ (self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq) → Iterator

__len__ (self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq) → int

__module__ = 'rti.connextdds'

__mul__ (self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: int) → rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq

__ne__ (self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq) → bool

__rmul__ (self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: int) → rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: int, arg1: rti.connextdds.PublicationBuiltinTopicData.DataReader)` -> None
2. `__setitem__(self: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, arg0: slice, arg1: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq)` -> None

Assign list elements using a slice object

append (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, *x*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → None

Add an item to the end of the list

clear (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq) → None

Clear the contents

count (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, *x*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, *L*: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq) -> None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, *i*: int, *x*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq) -> rti.connextdds.PublicationBuiltinTopicData.DataReader

Remove and return the last item

2. pop(*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, *i*: int) -> rti.connextdds.PublicationBuiltinTopicData.DataReader

Remove and return the item at index *i*

remove (*self*: rti.connextdds.PublicationBuiltinTopicData.DataReaderSeq, *x*: rti.connextdds.PublicationBuiltinTopicData.DataReader) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class DataWriter

Bases: *IEntity*, *IAnyDataWriter*

__enter__ (*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter) → *rti.connextdds.PublicationBuiltinTopicData.DataWriter*

Enter a context for this DataWriter, to be cleaned up on exiting context

__eq__ (*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg0*: rti.connextdds.PublicationBuiltinTopicData.DataWriter) → bool

Test for equality.

__exit__ (*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg0*: object, *arg1*: object, *arg2*: object) → None

Exit the context for this DataWriter, cleaning up resources.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, topic: rti.connextdds.PublicationBuiltinTopicData.Topic) -> None`

Creates a DataWriter in the implicit publisher with default QoS.

2. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, topic: rti.connextdds.PublicationBuiltinTopicData.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.PublicationBuiltinTopicData.DataWriterListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Creates a DataWriter in the implicit publisher with specific QoS and optionally a listener.

3. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.PublicationBuiltinTopicData.Topic) -> None`

Creates a DataWriter in a publisher with default QoS.

4. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.PublicationBuiltinTopicData.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.PublicationBuiltinTopicData.DataWriterListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Creates a DataWriter in a publisher with specific QoS and optionally a listener.

5. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, writer: rti.connextdds.IAnyDataWriter) -> None`

Create a typed DataWriter from an AnyDataWriter.

6. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, entity: rti.connextdds.IEntity) -> None`

Create a typed DataWriter from an Entity.

__lshift__ (*args, **kwargs)

Overloaded function.

1. `__lshift__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, arg0: rti.connextdds.DataWriterQos) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Sets the DataWriterQos.

2. `__lshift__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, arg0: Tuple[rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.Time]) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Writes a paired sample with a timestamp.

3. `__lshift__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, arg0: Tuple[rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.InstanceHandle]) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Writes a paired sample with an instance handle.

4. `__lshift__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, arg0: rti.connextdds.PublicationBuiltinTopicData.TimestampedSeq) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Writes a sequence of pairs of samples with timestamps.

5. `__lshift__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, arg0: rti.connextdds.PublicationBuiltinTopicDataSeq) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Writes a sequence of samples.

6. `__lshift__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, arg0: rti.connextdds.PublicationBuiltinTopicData) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Writes a sample.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, arg0: rti.connextdds.PublicationBuiltinTopicData.DataWriter) -> bool`

Test for inequality.

`__rshift__(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, arg0: rti.connextdds.DataWriterQos) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Get the DataWriterQos.

`assert_liveliness(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter) -> None`

Manually asserts the liveliness of the DataWriter.

`close(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter) -> None`

Close this DataWriter.

`create_data(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter) -> rti.connextdds.PublicationBuiltinTopicData`

Create data of the writer's associated type and initialize it.

`property datawriter_cache_status`

Get a copy of the cache status for this writer.

`property datawriter_protocol_status`

Get a copy of the protocol status for this writer.

`dispose_instance(*args, **kwargs)`

Overloaded function.

1. `dispose_instance(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Dispose an instance.

2. `dispose_instance(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Dispose an instance with a timestamp.

3. `dispose_instance(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> None`

Dispose an instance with params.

dispose_instance_async (*args, **kwargs)

Overloaded function.

1. `dispose_instance_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> object`

Dispose an instance.

2. `dispose_instance_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Dispose an instance with a timestamp.

3. `dispose_instance_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, key_holder: rti.connextdds.PublicationBuiltinTopicData) -> object`

Dispose the instance associated with key_holder.

4. `dispose_instance_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, key_holder: rti.connextdds.PublicationBuiltinTopicData, timestamp: rti.connextdds.Time) -> object`

Dispose the instance associated with key_holder using a timestamp

5. `dispose_instance_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> object`

Dispose an instance with params.

static find_all_by_topic (publisher: rti.connextdds.Publisher, topic_name: str) → *rti.connextdds.PublicationBuiltinTopicData.DataWriterSeq*

Retrieve all DataWriters for the given topic name in the publisher.

static find_by_name (*args, **kwargs)

Overloaded function.

1. `find_by_name(participant: rti.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.PublicationBuiltinTopicData.DataWriter]`

Find DataWriter in DomainParticipant with the provided name, returning the first found.

2. `find_by_name(publisher: rti.connextdds.Publisher, name: str) -> Optional[rti.connextdds.PublicationBuiltinTopicData.DataWriter]`

Find DataWriter in Publisher with the DataReader's name, returning the first found.

static find_by_topic (publisher: rti.connextdds.Publisher, name: str) → *Optional[rti.connextdds.PublicationBuiltinTopicData.DataWriter]*

Find DataWriter in publisher with a topic name, returning the first found.

flush (self: rti.connextdds.PublicationBuiltinTopicData.DataWriter) → None

Flushes the batch in progress in the context of the calling thread.

is_matched_subscription_active (self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, arg0: rti.connextdds.InstanceHandle) → bool

A boolean indicating whether or not the matched subscription is active.

is_sample_app_acknowledged (self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, sample_id: rti.connextdds.SampleIdentity) → bool

Indicates if a sample is considered application-acknowledged.

key_value (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.PublicationBuiltinTopicData*

Retrieve the instance key that corresponds to an instance handle.

property listener

Get the listener associated with the DataWriter or set the listener.

property liveliness_lost_status

Get a copy of the LivelinessLostStatus.

lookup_instance (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *key_holder*: rti.connexdds.PublicationBuiltinTopicData) → *rti.connexdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

matched_subscription_data (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.SubscriptionBuiltinTopicData*

Get the SubscriptionBuiltinTopicData for a subscription matched to this DataWriter.

matched_subscription_datawriter_protocol_status (*args, **kwargs)

Overloaded function.

1. `matched_subscription_datawriter_protocol_status(self: rti.connexdds.PublicationBuiltinTopicData.DataWriter, handle: rti.connexdds.InstanceHandle) -> rti.connexdds.DataWriterProtocolStatus`

Get a copy of the protocol status for this writer per a matched subscription handle.

2. `matched_subscription_datawriter_protocol_status(self: rti.connexdds.PublicationBuiltinTopicData.DataWriter, locator: rti.connexdds.Locator) -> rti.connexdds.DataWriterProtocolStatus`

Get a copy of the protocol status for this writer per a matched subscription locator.

matched_subscription_participant_data (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a subscription matched to this DataWriter.

property matched_subscriptions

Get a copy of the list of the currently matched subscription handles.

property matched_subscriptions_locators

The locators used to communicate with matched DataReaders.

property offered_deadline_missed_status

Get a copy of the OfferedDeadlineMissedStatus.

property offered_incompatible_qos_status

Get a copy of the OfferedIncompatibleQosStatus

property publication_matched_status

Get a copy of the PublicationMatchedStatus

property publisher

Get the Publisher that owns this DataWriter.

property qos

The DataWriterQos for this DataWriter. This property's getter returns a deep copy.

register_instance (*args, **kwargs)

Overloaded function.

1. register_instance(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, key_holder: rti.connextdds.PublicationBuiltinTopicData) -> rti.connextdds.InstanceHandle

Informs RTI Connext that the application will be modifying a particular instance.

2. register_instance(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, key_holder: rti.connextdds.PublicationBuiltinTopicData, timestamp: rti.connextdds.Time) -> rti.connextdds.InstanceHandle

Informs RTI Connext that the application will be modifying a particular instance and specified the timestamp.

3. register_instance(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, key_holder: rti.connextdds.PublicationBuiltinTopicData, params: rti.connextdds.WriteParams) -> rti.connextdds.InstanceHandle

Registers instance with parameters.

property reliable_reader_activity_changed_status

Get a copy of the reliable reader activity changed status for this writer.

property reliable_writer_cache_changed_status

Get a copy of the reliable cache status for this writer.

property service_request_accepted_status

Get a copy of the service request accepted status for this writer.

set_listener (self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, listener: rti.connextdds.PublicationBuiltinTopicData.DataWriterListener, event_mask: rti.connextdds.StatusMask) → None

Set the listener and event mask for the DataWriter.

property topic

Get the Topic object associated with this DataWriter.

property topic_name

Get the topic name associated with this DataWriter.

property type_name

Get the type name for the topic object associated with this DataWriter.

unregister_instance (*args, **kwargs)

Overloaded function.

1. `unregister_instance(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Unregister an instance.

2. `unregister_instance(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.PublicationBuiltinTopicData.DataWriter`

Unregister an instance with timestamp.

3. `unregister_instance(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> None`

Unregister an instance with parameters.

`unregister_instance_async` (**args, **kwargs*)

Overloaded function.

1. `unregister_instance_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> object`

Unregister an instance.

2. `unregister_instance_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Unregister an instance with timestamp.

3. `unregister_instance_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, key_holder: rti.connextdds.PublicationBuiltinTopicData) -> object`

Unregister the instance associated with `key_holder`.

4. `unregister_instance_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, key_holder: rti.connextdds.PublicationBuiltinTopicData, timestamp: rti.connextdds.Time) -> object`

Unregister the instance associate with `key_holder` using a timestamp.

5. `unregister_instance_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> object`

Unregister an instance with parameters.

`wait_for_acknowledgments` (*self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, max_wait: rti.connextdds.Duration*) → None

Blocks the calling thread until all data written by a reliable DataWriter is acknowledged or until the timeout expires.

`wait_for_asynchronous_publishing` (*self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, max_wait: rti.connextdds.Duration*) → None

This operation blocks the calling thread (up to `max_wait`) until all data written by the asynchronous DataWriter is sent and acknowledged (if reliable) by all matched DataReader entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that `max_wait` elapsed before all the data was sent and/or acknowledged.

In other words, this guarantees that sending to best effort DataReader is complete in addition to what `DataWriter.wait_for_acknowledgments()` provides.

If the DataWriter does not have PublishMode kind set to PublishModeKind.ASYNCHRONOUS the operation will complete immediately

wait_for_asynchronous_publishing_async (*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *max_wait*: rti.connextdds.Duration) → object

This function is awaitable until either a timeout of *max_wait* or all data written by the asynchronous DataWriter is sent and acknowledged (if reliable) by all matched DataReader entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that *max_wait* elapsed before all the data was sent and/or acknowledged. This function works with asyncio.

In other words, this guarantees that sending to best effort DataReader is complete in addition to what DataWriter.wait_for_acknowledgments() provides.

If the DataWriter does not have PublishMode kind set to PublishModeKind.ASYNCHRONOUS the operation will complete immediately

write (**args*, ***kwargs*)

Overloaded function.

1. write(*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *samples*: rti.connextdds.PublicationBuiltinTopicDataSeq) → None

Write a sequence of samples.

2. write(*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *samples*: rti.connextdds.PublicationBuiltinTopicDataSeq, *timestamp*: rti.connextdds.Time) → None

Write a sequence of samples with a timestamp.

3. write(*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *sample*: rti.connextdds.PublicationBuiltinTopicData) → None

Write a sample.

4. write(*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *sample*: rti.connextdds.PublicationBuiltinTopicData, *timestamp*: rti.connextdds.Time) → None

Write a sample with a specified timestamp.

5. write(*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *sample*: rti.connextdds.PublicationBuiltinTopicData, *handle*: rti.connextdds.InstanceHandle) → None

Write a sample with an instance handle.

6. write(*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *sample*: rti.connextdds.PublicationBuiltinTopicData, *handle*: rti.connextdds.InstanceHandle, *timestamp*: rti.connextdds.Time) → None

Write a sample with an instance handle and specified timestamp.

7. write(*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *sample*: rti.connextdds.PublicationBuiltinTopicData, *params*: rti.connextdds.WriteParams) → None

Write with advanced parameters.

write_async (**args*, ***kwargs*)

Overloaded function.

1. write_async(*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *sample*: rti.connextdds.PublicationBuiltinTopicData) → object

Write a sample. This method is awaitable and is only for use with asyncio.

2. write_async(*self*: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *sample*: rti.connextdds.PublicationBuiltinTopicData, *timestamp*: rti.connextdds.Time) → object

Write a sample with a specified timestamp. This methods is awaitable and only for use with asyncio.

```
3. write_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, sample:
   rti.connextdds.PublicationBuiltinTopicData, handle: rti.connextdds.InstanceHandle) ->
   object
```

Write a sample with an instance handle. This method is awaitable and only for use with asyncio.

```
4. write_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, sample:
   rti.connextdds.PublicationBuiltinTopicData, handle: rti.connextdds.InstanceHandle,
   timestamp: rti.connextdds.Time) -> object
```

Write a sample with an instance handle and specified timestamp. This method is awaitable and only for use with asyncio.

```
5. write_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, samples:
   rti.connextdds.PublicationBuiltinTopicDataSeq) -> object
```

Write a sequence of samples. This method is awaitable and only for use with asyncio.

```
6. write_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, samples:
   rti.connextdds.PublicationBuiltinTopicDataSeq, timestamp: rti.connextdds.Time) ->
   object
```

Write a sequence of samples with a timestamp. This method is awaitable and only for use with asyncio.

```
7. write_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, samples:
   rti.connextdds.PublicationBuiltinTopicDataSeq, handles: rti.connextdds.InstanceHan-
   dleSeq) -> object
```

Write a sequence of samples with their instance handles. This method is awaitable and only for use with asyncio.

```
8. write_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, samples:
   rti.connextdds.PublicationBuiltinTopicDataSeq, handles: rti.connextdds.InstanceHan-
   dleSeq, timestamp: rti.connextdds.Time) -> object
```

Write a sequence of samples with their instance handles and a timestamp. This method is awaitable and only for use with asyncio.

```
9. write_async(self: rti.connextdds.PublicationBuiltinTopicData.DataWriter, sample:
   rti.connextdds.PublicationBuiltinTopicData, params: rti.connextdds.WriteParams) ->
   object
```

Write with advanced parameters.

class DataWriterListener

Bases: pybind11_object

```
__init__ (self: rti.connextdds.PublicationBuiltinTopicData.DataWriterListener) → None
```

```
__module__ = 'rti.connextdds'
```

```
on_application_acknowledgment (self: rti.connextdds.PublicationBuiltinTopic-
   Data.DataWriterListener, arg0:
   rti.connextdds.PublicationBuiltinTopic-
   Data.DataWriter, arg1:
   rti.connextdds.AcknowledgmentInfo) → None
```

On application acknowledgment callback

on_instance_replaced (*self*:
 rti.connextdds.PublicationBuiltinTopicData.DataWriterListener,
arg0: rti.connextdds.PublicationBuiltinTopicData.DataWriter,
arg1: rti.connextdds.InstanceHandle) → None

On instance replaced callback.

on_liveliness_lost (*self*:
 rti.connextdds.PublicationBuiltinTopicData.DataWriterListener,
arg0: rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1*:
 rti.connextdds.LivelinessLostStatus) → None

Liveliness lost callback.

on_offered_deadline_missed (*self*: rti.connextdds.PublicationBuiltinTopic-
 Data.DataWriterListener, *arg0*:
 rti.connextdds.PublicationBuiltinTopicData.DataWriter,
arg1: rti.connextdds.OfferedDeadlineMissedStatus) →
 None

Offered deadline missed callback.

on_offered_incompatible_qos (*self*: rti.connextdds.PublicationBuiltinTopic-
 Data.DataWriterListener, *arg0*:
 rti.connextdds.PublicationBuiltinTopic-
 Data.DataWriter, *arg1*:
 rti.connextdds.OfferedIncompatibleQosStatus) →
 None

Offered incompatible QoS callback.

on_publication_matched (*self*: rti.connextdds.PublicationBuiltinTopic-
 Data.DataWriterListener, *arg0*:
 rti.connextdds.PublicationBuiltinTopicData.DataWriter, *arg1*:
 rti.connextdds.PublicationMatchedStatus) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connextdds.PublicationBuiltin-
 TopicData.DataWriterListener, *arg0*:
 rti.connextdds.PublicationBuiltinTopic-
 Data.DataWriter, *arg1*:
 rti.connextdds.ReliableReaderActivity-
 ChangedStatus) →
 None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connextdds.PublicationBuiltinTopic-
 Data.DataWriterListener, *arg0*:
 rti.connextdds.PublicationBuiltinTopic-
 Data.DataWriter, *arg1*:
 rti.connextdds.ReliableWriterCacheChanged-
 Status) →
 None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class DataWriterSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) → *rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq*

__bool__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.DataWriter) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: int) → None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: slice) → None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *s*: slice) → rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: int) → rti.connexdds.PublicationBuiltinTopicData.DataWriter

__hash__ = None

__iadd__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) → *rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq*

__imul__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: int) → *rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) -> None
2. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) → Iterator

__len__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: int) → *rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq*

__ne__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) → bool

__rmul__ (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: int) → *rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: int, *arg1*: rti.connexdds.PublicationBuiltinTopicData.DataWriter) -> None
2. **__setitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *arg0*: slice, *arg1*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.DataWriter) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) → None

Clear the contents

count (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.DataWriter) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *L*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) -> None

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *i*: int, *x*: rti.connexdds.PublicationBuiltinTopicData.DataWriter) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq) -> rti.connexdds.PublicationBuiltinTopicData.DataWriter`

Remove and return the last item

2. `pop(self: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, i: int) -> rti.connexdds.PublicationBuiltinTopicData.DataWriter`

Remove and return the item at index *i*

remove (*self*: rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.DataWriter) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class ITopicDescription

Bases: *IEntity*

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

property name

The name of the entity conforming to the ITopicDescription interface.

property participant

The parent DomainParticipant.

property type_name

The name of the associated type.

class LoanedSample

Bases: *pybind11_object*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.PublicationBuiltinTopicData.LoanedSample) -> None`

Basic constructor

2. `__init__(self: rti.connexdds.PublicationBuiltinTopicData.LoanedSample, data: rti.connexdds.PublicationBuiltinTopicData, info: rti.connexdds.SampleInfo) -> None`

Construct LoanedSample with data and info.

__iter__ (*self*: rti.connexdds.PublicationBuiltinTopicData.LoanedSample) → object

```
__module__ = 'rti.connexdds'
```

property data

Get the data associated with the sample.

property info

Get the info associated with the sample.

class LoanedSamples

Bases: *pybind11_object*

```
__enter__ (self: rti.connexdds.PublicationBuiltinTopicData.LoanedSamples) →  
           rti.connexdds.PublicationBuiltinTopicData.LoanedSamples
```

Enter a context for the loaned samples, loan returned on context exit.

```
__exit__ (self: rti.connexdds.PublicationBuiltinTopicData.LoanedSamples, arg0: object,  
          arg1: object, arg2: object) → None
```

Exit the context for the loaned samples, returning the resources.

```
__getitem__ (self: rti.connexdds.PublicationBuiltinTopicData.LoanedSamples, arg0: int)  
            → rti.connexdds.PublicationBuiltinTopicData.LoanedSample
```

Access a LoanedSample object in an array-like syntax

```
__init__ (self: rti.connexdds.PublicationBuiltinTopicData.LoanedSamples) → None
```

Create an empty LoanedSamples object.

```
__iter__ (self: rti.connexdds.PublicationBuiltinTopicData.LoanedSamples) → Iterator
```

```
__len__ (self: rti.connexdds.PublicationBuiltinTopicData.LoanedSamples) → int
```

Get the number of samples in the loan.

```
__module__ = 'rti.connexdds'
```

property length

Get the number of samples in the loan.

```
return_loan (self: rti.connexdds.PublicationBuiltinTopicData.LoanedSamples) → None
```

Returns the loan to the DataReader.

class NoOpDataReaderListener

Bases: *DataReaderListener*

```
__init__ (self: rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener) →  
          None
```

```
__module__ = 'rti.connexdds'
```

```
on_data_available (self: rti.connexdds.PublicationBuiltinTopicData.NoOpDataRead-  
                    erListener, arg0:  
                    rti.connexdds.PublicationBuiltinTopicData.DataReader) → None
```

Data available callback.

on_liveliness_changed (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connexdds.LivelinessChangedStatus) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SampleLostStatus) → None

Sample lost callback.

on_sample_rejected (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class NoOpDataWriterListener

Bases: *DataWriterListener*

__init__ (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener) → None

__module__ = 'rti.connexdds'

on_application_acknowledgment (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.AcknowledgmentInfo) → None

On application acknowledgment callback

on_instance_replaced (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.InstanceHandle) → None

On instance replaced callback.

on_liveliness_lost (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.LivelinessLostStatus) → None

Liveliness lost callback.

on_offered_deadline_missed (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

on_offered_incompatible_qos (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

on_publication_matched (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.PublicationMatchedStatus) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class NoOpTopicListener

Bases: *TopicListener*

__init__ (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpTopicListener) → None

__module__ = 'rti.connexdds'

on_inconsistent_topic (*self*: rti.connexdds.PublicationBuiltinTopicData.NoOpTopicListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.Topic, *arg1*: rti.connexdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class Sample

Bases: *pybind11_object*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.Sample, *data*: rti.connexdds.PublicationBuiltinTopicData, *info*: rti.connexdds.SampleInfo) -> None

Construct Sample with data and info.

2. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.Sample, *sample*: rti.connexdds.PublicationBuiltinTopicData.Sample) -> None

Copy constructor.

3. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.Sample) -> None

Basic constructor

4. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.Sample, *loaned_sample*: rti.connexdds.PublicationBuiltinTopicData.LoanedSample) -> None

Construct a sample with a loaned sample.

__iter__ (*self*: rti.connexdds.PublicationBuiltinTopicData.Sample) → object

```
__module__ = 'rti.connextdds'
```

property data

The data associated with the sample.

property info

Get the info associated with the sample.

class SharedSamples

Bases: *pybind11_object*

```
__getitem__ (self: rti.connextdds.PublicationBuiltinTopicData.SharedSamples, arg0: int) →  
rti.connextdds.PublicationBuiltinTopicData.LoanedSample
```

Get the sample at the specified index.

```
__init__ (self: rti.connextdds.PublicationBuiltinTopicData.SharedSamples, loaned_samples:  
rti.connextdds.PublicationBuiltinTopicData.LoanedSamples) → None
```

Constructs an instance of SharedSamples, removing ownership of the loan from the Loaned Samples.

```
__iter__ (self: rti.connextdds.PublicationBuiltinTopicData.SharedSamples) → Iterator
```

Make a sample iterator

```
__len__ (self: rti.connextdds.PublicationBuiltinTopicData.SharedSamples) → int
```

Returns the number of samples.

```
__module__ = 'rti.connextdds'
```

class Topic

Bases: *ITopicDescription, IAnyTopic*

```
__eq__ (self: rti.connextdds.PublicationBuiltinTopicData.Topic, arg0:  
rti.connextdds.PublicationBuiltinTopicData.Topic) → bool
```

Test for equality.

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.Topic, entity: rti.connextdds.IEntity) -> None`

Cast an Entity to a Topic.

2. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.Topic, topic_description: rti.connextdds.PublicationBuiltinTopicData.ITopicDescription) -> None`

Cast an ITopicDescription to a Topic.

3. `__init__(self: rti.connextdds.PublicationBuiltinTopicData.Topic, topic: rti.connextdds.IAnyTopic) -> None`

Create a typed Topic from an AnyTopic.

```
__module__ = 'rti.connextdds'
```

__ne__ (*self*: rti.connexdds.PublicationBuiltinTopicData.Topic, *arg0*: rti.connexdds.PublicationBuiltinTopicData.Topic) → bool

Test for inequality.

static find (*participant*: rti.connexdds.DomainParticipant, *name*: str) → Optional[rti.connexdds.PublicationBuiltinTopicData.Topic]

Look up a Topic by its name in the DomainParticipant.

property inconsistent_topic_status

Get a copy of the current InconsistentTopicStatus for this Topic.

property listener

The listener.

property qos

Get the TopicQos for this Topic.

This property's getter returns a deep copy.

set_listener (*self*: rti.connexdds.PublicationBuiltinTopicData.Topic, *listener*: rti.connexdds.PublicationBuiltinTopicData.TopicListener, *event_mask*: rti.connexdds.StatusMask) → None

Set the listener and event mask.

class TopicDescription

Bases: *ITopicDescription*

__eq__ (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicDescription, *arg0*: rti.connexdds.PublicationBuiltinTopicData.TopicDescription) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicDescription, *topic*: rti.connexdds.PublicationBuiltinTopicData.ITopicDescription) → None

Cast a Topic to a TopicDescription.

2. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicDescription, *entity*: rti.connexdds.IEntity) → None

Cast a Topic to a TopicDescription.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicDescription, *arg0*: rti.connexdds.PublicationBuiltinTopicData.TopicDescription) → bool

Test for inequality.

class TopicListener

Bases: *pybind11_object*

`__init__` (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicListener) → None

`__module__` = 'rti.connexdds'

`on_inconsistent_topic` (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicListener, *arg0*: rti.connexdds.PublicationBuiltinTopicData.Topic, *arg1*: rti.connexdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class TopicSeq

Bases: pybind11_object

`__add__` (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) → *rti.connexdds.PublicationBuiltinTopicData.TopicSeq*

`__bool__` (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) → bool
Check whether the list is nonempty

`__contains__` (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.Topic) → bool

Return true the container contains *x*

`__delitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__delitem__`(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: int) → None

Delete the list elements at index *i*

2. `__delitem__`(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: slice) → None

Delete list elements using a slice object

`__eq__` (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) → bool

`__getitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__`(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *s*: slice) → *rti.connexdds.PublicationBuiltinTopicData.TopicSeq*

Retrieve list elements using a slice object

2. `__getitem__`(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: int) → *rti.connexdds.PublicationBuiltinTopicData.Topic*

`__hash__` = None

`__iadd__` (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) → *rti.connexdds.PublicationBuiltinTopicData.TopicSeq*

__imul__ (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: int) → *rti.connexdds.PublicationBuiltinTopicData.TopicSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) -> None
2. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) → Iterator

__len__ (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: int) → *rti.connexdds.PublicationBuiltinTopicData.TopicSeq*

__ne__ (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) → bool

__rmul__ (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: int) → *rti.connexdds.PublicationBuiltinTopicData.TopicSeq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: int, *arg1*: rti.connexdds.PublicationBuiltinTopicData.Topic) -> None
2. **__setitem__**(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *arg0*: slice, *arg1*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.Topic) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) → None

Clear the contents

count (*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *x*: rti.connexdds.PublicationBuiltinTopicData.Topic) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq, *L*: rti.connexdds.PublicationBuiltinTopicData.TopicSeq) -> None

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *i*: int, *x*: rti.connextdds.PublicationBuiltinTopicData.Topic) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq) -> rti.connextdds.PublicationBuiltinTopicData.Topic`

Remove and return the last item

2. `pop(self: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, i: int) -> rti.connextdds.PublicationBuiltinTopicData.Topic`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.PublicationBuiltinTopicData.TopicSeq, *x*: rti.connextdds.PublicationBuiltinTopicData.Topic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class ValidLoanedSamples

Bases: `pybind11_object`

__enter__ (*self*: rti.connextdds.PublicationBuiltinTopicData.ValidLoanedSamples) → *rti.connextdds.PublicationBuiltinTopicData.ValidLoanedSamples*

__exit__ (*self*: rti.connextdds.PublicationBuiltinTopicData.ValidLoanedSamples, *arg0*: object, *arg1*: object, *arg2*: object) → None

__init__ (*args, **kwargs)

__iter__ (*self*: rti.connextdds.PublicationBuiltinTopicData.ValidLoanedSamples) → Iterator

__module__ = 'rti.connextdds'

class WriterContentFilter

Bases: `ContentFilter`

__init__ (*self*: rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter) → None

__module__ = 'rti.connextdds'

writer_attach (*self*: rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter) → Optional[object]

A writer-side filtering API to create some state that can facilitate filtering on the writer side.

writer_compile (*self*: rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object], *property*: rti.connextdds.ExpressionProperty, *expression*: str, *parameters*: rti.connextdds.StringSeq, *type_code*: Optional[rti.connextdds.DynamicType], *type_class_name*: str, *cookie*: rti.connextdds.Cookie) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

writer_detach (*self*: rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter, *writer_filter_data*: *Optional[object]*) → None

A writer-side filtering API to clean up a previously created state using writer_attach.

writer_evaluate (*self*: rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter, *writer_filter_data*: *Optional[object]*, *sample*: rti.connextdds.PublicationBuiltinTopicData, *meta_data*: rti.connextdds.FilterSampleInfo) → *rti.connextdds.CookieVector*

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

writer_finalize (*self*: rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter, *writer_filter_data*: *Optional[object]*, *cookie*: rti.connextdds.Cookie) → None

A writer-side filtering API to clean up a previously compiled instance of the content filter.

writer_return_loan (*self*: rti.connextdds.PublicationBuiltinTopicData.WriterContentFilter, *writer_filter_data*: *Optional[object]*, *cookies*: rti.connextdds.CookieVector) → None

A writer-side filtering API to return the loan on the list of DataReaders returned by writer_evaluate.

class WriterContentFilterHelper

Bases: *WriterContentFilter*

__init__ (*self*: rti.connextdds.PublicationBuiltinTopicData.WriterContentFilterHelper) → None

__module__ = 'rti.connextdds'

add_cookie (*self*: rti.connextdds.PublicationBuiltinTopicData.WriterContentFilterHelper, *cookie*: rti.connextdds.Cookie) → None

A helper function which will add a Cookie to the Cookie sequence that is then returned by the writer_evaluate function.

writer_evaluate_helper (*self*: rti.connextdds.PublicationBuiltinTopicData.WriterContentFilterHelper, *writer_filter_data*: *Optional[object]*, *sample*: rti.connextdds.PublicationBuiltinTopicData, *meta_data*: rti.connextdds.FilterSampleInfo) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

__eq__ (*self*: rti.connextdds.PublicationBuiltinTopicData, *arg0*: rti.connextdds.PublicationBuiltinTopicData) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.PublicationBuiltinTopicData) → None
Create a default PublicationBuiltinTopicData.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.PublicationBuiltinTopicData, *arg0*:
rti.connexdds.PublicationBuiltinTopicData) → bool
Test for inequality.

builtin_topic_name = 'DCPSPublication'

property data_tag

The DataTag policy of the corresponding DataWriter.

property deadline

The Deadline policy of the corresponding DataWriter.

property destination_order

The DestinationOrder policy of the corresponding DataWriter.

property disable_positive_acks

See whether or not a matching DataReader will send positive acknowledgments for reliability.

property durability

The Durability policy of the corresponding DataWriter.

property durability_service

The DurabilityService policy of the corresponding DataWriter.

property group_data

The GroupData policy of the corresponding DataWriter's Publisher.

property key

The DCPS key to distinguish entries.

property latency_budget

The LatencyBudget policy of the corresponding DataWriter.

property lifespan

The Lifespan policy of the corresponding DataWriter.

property liveliness

The Liveliness policy of the corresponding DataWriter.

property locator_filter

The locator filters of the corresponding DataWriter.

property ownership

The Ownership policy of the corresponding DataWriter.

property ownership_strength

The OwnershipStrength policy of the corresponding DataWriter.

property participant_key

The DCPS key of the DomainParticipant to which the DataWriter belongs.

property partition

The Partition policy of the corresponding DataWriter's Publisher.

property presentation

The Presentation policy of the corresponding DataWriter's Publisher.

property product_version

The current version for RTI Connex.

property property

The propagated name-value properties of the corresponding DataWriter.

property publication_name

The publication name and role name.

property publisher_key

The DCPS key of the Publisher to which the DataWriter belongs.

property reliability

The Reliability policy of the corresponding DataWriter.

property representation

The Representation policy of the corresponding DataWriter.

property rtps_protocol_version

The version number of the RTPS wire protocol used.

property rtps_vendor_id

The ID of the vendor implementing the RTPS wire protocol.

property service

The Service policy

property topic_data

The TopicData policy of the corresponding DataWriter's Topic.

property topic_name

The name of the related Topic.

property type

The type.

property type_name

The name of the type attached to the Topic.

property unicast_locators

The custom unicast locators that the endpoint can specify.

property user_data

The UserData policy of the corresponding DataWriter.

property virtual_guid

The virtual Guid associated to the DataWriter.

class rti.connexdds.PublicationBuiltinTopicDataSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicDataSeq) → *rti.connexdds.PublicationBuiltinTopicDataSeq*

__bool__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *x*: rti.connexdds.PublicationBuiltinTopicData) → bool
Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicDataSeq) → bool

__getitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *s*: slice) -> rti.connexdds.PublicationBuiltinTopicDataSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: int) -> rti.connexdds.PublicationBuiltinTopicData

__hash__ = None

__iadd__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicDataSeq) → *rti.connexdds.PublicationBuiltinTopicDataSeq*

__imul__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: int) →
rti.connexdds.PublicationBuiltinTopicDataSeq

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicDataSeq) -> None
2. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: rti.connexdds.PublicationBuiltinTopicDataSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq) → Iterator

__len__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: int) →
rti.connexdds.PublicationBuiltinTopicDataSeq

__ne__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*:
rti.connexdds.PublicationBuiltinTopicDataSeq) → bool

__rmul__ (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: int) →
rti.connexdds.PublicationBuiltinTopicDataSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: int, *arg1*: rti.connexdds.PublicationBuiltinTopicData) -> None
2. **__setitem__**(*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *arg0*: slice, *arg1*: rti.connexdds.PublicationBuiltinTopicDataSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *x*:
rti.connexdds.PublicationBuiltinTopicData) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq) → None

Clear the contents

count (*self*: rti.connexdds.PublicationBuiltinTopicDataSeq, *x*:
rti.connexdds.PublicationBuiltinTopicData) → int

Return the number of times *x* appears in the list

extend (*args, **kwargs)

Overloaded function.

1. extend(self: rti.connextdds.PublicationBuiltinTopicDataSeq, L: rti.connextdds.PublicationBuiltinTopicDataSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.PublicationBuiltinTopicDataSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.PublicationBuiltinTopicDataSeq, i: int, x: rti.connextdds.PublicationBuiltinTopicData) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connextdds.PublicationBuiltinTopicDataSeq) -> rti.connextdds.PublicationBuiltinTopicData

Remove and return the last item

2. pop(self: rti.connextdds.PublicationBuiltinTopicDataSeq, i: int) -> rti.connextdds.PublicationBuiltinTopicData

Remove and return the item at index i

remove (self: rti.connextdds.PublicationBuiltinTopicDataSeq, x: rti.connextdds.PublicationBuiltinTopicData) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq

Bases: pybind11_object

__add__ (self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) → rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq

__bool__ (self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) → bool
Check whether the list is nonempty

__contains__ (self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, x: Tuple[rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.Time]) → bool

Return true the container contains x

__delitem__ (*args, **kwargs)

Overloaded function.

1. __delitem__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: int) -> None

Delete the list elements at index i

2. `__delitem__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: slice)`
-> None

Delete list elements using a slice object

`__eq__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) → bool`

`__getitem__(*args, **kwargs)`

Overloaded function.

1. `__getitem__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, s: slice) -> rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: int) -> Tuple[rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.Time]`

`__hash__ = None`

`__iadd__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) → rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq`

`__imul__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: int) → rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) -> None`
2. `__init__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) → Iterator`

`__len__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) → int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: int) → rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq`

`__ne__(self: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq) → bool`

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` = <capsule object NULL>

`__rmul__` (*self*: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq, *arg0*: int) → rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq

`__setitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__setitem__(self: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: int, arg1: Tuple[rti.connexdds.PublicationBuiltinTopicData, rti.connexdds.Time])` -> None
2. `__setitem__(self: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq, arg0: slice, arg1: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq)` -> None

Assign list elements using a slice object

`append` (*self*: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq, *x*: Tuple[rti.connexdds.PublicationBuiltinTopicData, rti.connexdds.Time]) → None

Add an item to the end of the list

`clear` (*self*: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq) → None

Clear the contents

`count` (*self*: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq, *x*: Tuple[rti.connexdds.PublicationBuiltinTopicData, rti.connexdds.Time]) → int

Return the number of times *x* appears in the list

`extend` (**args*, ***kwargs*)

Overloaded function.

1. `extend(self: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq, L: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq)` -> None

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq, L: Iterable)` -> None

Extend the list by appending all the items in the given list

`insert` (*self*: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq, *i*: int, *x*: Tuple[rti.connexdds.PublicationBuiltinTopicData, rti.connexdds.Time]) → None

Insert an item at a given position.

`pop` (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq)` -> Tuple[rti.connexdds.PublicationBuiltinTopicData, rti.connexdds.Time]

Remove and return the last item

2. `pop(self: rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq, i: int)` -> Tuple[rti.connexdds.PublicationBuiltinTopicData, rti.connexdds.Time]

Remove and return the item at index *i*

remove (*self*: rti.connextdds.PublicationBuiltinTopicDataTimestampedSeq, *x*:
Tuple[rti.connextdds.PublicationBuiltinTopicData, rti.connextdds.Time]) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connextdds.**PublicationMatchedStatus**

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connextdds'

property **current_count**

The number of DataReaders that are currently matched with this DataWriter.

property **current_count_change**

The delta for the current count since the last time the listener fired or the status was read.

property **current_count_peak**

The highest value that the current count has reached.

property **last_subscription_handle**

A handle to the DataReader that caused the most recent change to to this status.

property **total_count**

Total count of times the DataWriter matched with a DataReader.

property **total_count_change**

The delta in the total count since the last time the listener fired or the status was read.

class rti.connextdds.**PublishMode**

Bases: pybind11_object

PUBLICATION_PRIORITY_UNDEFINED = 0

__eq__ (*self*: rti.connextdds.PublishMode, *arg0*: rti.connextdds.PublishMode) → bool
 Test for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.PublishMode) → None
 Creates a PublishMode qos policy of synchronous kind.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.PublishMode, *arg0*: rti.connextdds.PublishMode) → bool
 Test for inequality.

static **asynchronous** (**args*, ***kwargs*)

Overloaded function.

1. asynchronous() -> rti.connextdds.PublishMode

Creates a PublishMode qos policy of asynchronous kind with default flow controller and undefined priority.

2. asynchronous(flow_controller_name: str) -> rti.connextdds.PublishMode

Creates a PublishMode qos policy of asynchronous kind with a specific flow controller and undefined priority.

3. asynchronous(flow_controller_name: str, priority: int) -> rti.connextdds.PublishMode

Creates a PublishMode qos policy of asynchronous kind with specific flow controller and priority.

property flow_controller_name

The flow controller name associated to a DataWriter.

property kind

The publish mode (synchronous or asynchronous) for a DataWriter.

property priority

The priority of all samples written by a DataWriter.

synchronous = <rti.connextdds.PublishMode object>

class rti.connextdds.PublishModeKind

Bases: pybind11_object

ASYNCHRONOUS = <PublishModeKind.ASYNCHRONOUS: 1>

class PublishModeKind

Bases: pybind11_object

Members:

SYNCHRONOUS : Indicates to send data synchronously.

If DataWriterProtocol.push_on_write is true, data is sent immediately in the context of DataWriter.write().

As data is sent immediately in the context of the user thread, no flow control is applied.

ASYNCHRONOUS : Indicates to send data asynchronously.

Configures the DataWriter to delegate the task of data transmission to a separate publishing thread. The DataWriter.write() call does not send the data, but instead schedules the data to be sent later by its associated Publisher.

Each Publisher uses its dedicated publishing thread (AsynchronousPublisher) to send data for all its asynchronous DataWriters. For each asynchronous DataWriter, the associated FlowController determines when the publishing thread is allowed to send the data.

DataWriter.wait_for_asynchronous_publishing and Publisher.wait_for_asynchronous_publishing enable you to determine when the data has actually been sent.

ASYNCHRONOUS = <PublishModeKind.ASYNCHRONOUS: 1>

SYNCHRONOUS = <PublishModeKind.SYNCHRONOUS: 0>

__eq__ (*self: object, other: object*) → bool

__getstate__ (*self: object*) → int

__hash__ (*self: object*) → int

__index__ (*self: rti.connextdds.PublishModeKind.PublishModeKind*) → int

__init__ (*self: rti.connextdds.PublishModeKind.PublishModeKind, value: int*) → None

__int__ (*self: rti.connextdds.PublishModeKind.PublishModeKind*) → int

__members__ = {'ASYNCHRONOUS': <PublishModeKind.ASYNCHRONOUS: 1>, 'SYNCHRONOUS': <PublishModeKind.SYNCHRONOUS: 0>}

__module__ = 'rti.connextdds'

__ne__ (*self: object, other: object*) → bool

__repr__ (*self: object*) → str

__setstate__ (*self: rti.connextdds.PublishModeKind.PublishModeKind, state: int*) → None

__str__ ()

name(self: handle) -> str

property name

property value

SYNCHRONOUS = <PublishModeKind.SYNCHRONOUS: 0>

__eq__ (*self: rti.connextdds.PublishModeKind, arg0: rti.connextdds.PublishModeKind*) → bool
Apply operator to underlying enumerated values.

__ge__ (*self: rti.connextdds.PublishModeKind, arg0: rti.connextdds.PublishModeKind*) → bool
Apply operator to underlying enumerated values.

__gt__ (*self: rti.connextdds.PublishModeKind, arg0: rti.connextdds.PublishModeKind*) → bool
Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args, **kwargs*)

Overloaded function.

1. **__init__**(self: rti.connextdds.PublishModeKind) -> None

Initializes enum to 0.

2. **__init__**(self: rti.connextdds.PublishModeKind, arg0: rti.connextdds.PublishModeKind) -> None

Copy constructor.

```
__int__ (self: rti.connextdds.PublishModeKind) →
    rti.connextdds.PublishModeKind.PublishModeKind
```

Int conversion.

```
__le__ (self: rti.connextdds.PublishModeKind, arg0: rti.connextdds.PublishModeKind) → bool
    Apply operator to underlying enumerated values.
```

```
__lt__ (self: rti.connextdds.PublishModeKind, arg0: rti.connextdds.PublishModeKind) → bool
    Apply operator to underlying enumerated values.
```

```
__module__ = 'rti.connextdds'
```

```
__ne__ (self: rti.connextdds.PublishModeKind, arg0: rti.connextdds.PublishModeKind) → bool
    Apply operator to underlying enumerated values.
```

```
__str__ (self: rti.connextdds.PublishModeKind) → str
    String conversion.
```

property underlying

Retrieves the actual enumerated value.

class rti.connextdds.Publisher

Bases: *IEntity*

```
__eq__ (self: rti.connextdds.Publisher, arg0: rti.connextdds.Publisher) → bool
    Test for equality.
```

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connextdds.Publisher, participant: rti.connextdds.DomainParticipant) -> None`

Create a publisher.

2. `__init__(self: rti.connextdds.Publisher, participant: rti.connextdds.DomainParticipant, qos: rti.connextdds.PublisherQos, listener: rti.connextdds.PublisherListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Create a Publisher with the desired QoS policies and specified listener

3. `__init__(self: rti.connextdds.Publisher, entity: rti.connextdds.IEntity) -> None`

Cast an Entity to a Publisher.

```
__lshift__ (self: rti.connextdds.Publisher, arg0: rti.connextdds.PublisherQos) →
    rti.connextdds.Publisher
```

Set the PublisherQos.

```
__module__ = 'rti.connextdds'
```

__ne__ (*self*: rti.connexdds.Publisher, *arg0*: rti.connexdds.Publisher) → bool

Test for inequality.

__rshift__ (*self*: rti.connexdds.Publisher, *arg0*: rti.connexdds.PublisherQos) →
rti.connexdds.Publisher

Copy the PublisherQos.

property default_datawriter_qos

The default DataWriterQos.

This property's getter returns a deep copy.

find_datawriter (*self*: rti.connexdds.Publisher, *name*: str) →
Optional[*rti.connexdds.AnyData Writer*]

Find a DataWriter in this Publisher by its name.

find_datawriter_by_topic_name (*self*: rti.connexdds.Publisher, *topic_name*: str) →
Optional[*rti.connexdds.AnyData Writer*]

Find a DataWriter in this Publisher by its topic name. If more than one exists for this Topic, the first one found is returned.

find_datawriters (*self*: rti.connexdds.Publisher) → *rti.connexdds.AnyData WriterSeq*

Find all DataWriters in the Publisher.

property listener

Get/set the listener.

property participant

Get the parent DomainParticipant of this Publisher.

property qos

The PublisherQos for this Publisher.

This property's getter returns a deep copy.

set_listener (*self*: rti.connexdds.Publisher, *listener*: rti.connexdds.PublisherListener,
event_mask: rti.connexdds.StatusMask) → None

Bind the listener and event mask to the Publisher.

wait_for_acknowledgments (*self*: rti.connexdds.Publisher, *max_wait*:
rti.connexdds.Duration) → None

Blocks until all data written by this Publisher's reliable DataWriters is acknowledged or the timeout expires.

wait_for_asynchronous_publishing (*self*: rti.connexdds.Publisher, *max_wait*:
rti.connexdds.Duration) → None

Blocks until asynchronous sending is complete or timeout expires.

class rti.connexdds.PublisherListener

Bases: *AnyDataWriterListener*

`__init__` (*self*: rti.connexdds.PublisherListener) → None

`__module__` = 'rti.connexdds'

class rti.connexdds.PublisherQos

Bases: pybind11_object

`__eq__` (*self*: rti.connexdds.PublisherQos, *arg0*: rti.connexdds.PublisherQos) → bool

Test for equality

`__hash__` = None

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__`(*self*: rti.connexdds.PublisherQos) -> None

Create a PublisherQos with the default value for each policy.

2. `__init__`(*self*: rti.connexdds.PublisherQos, *publisher*: rti.connexdds.Publisher) -> None

Create a PublisherQos with the same settings as those applied to the provided Publisher object.

3. `__init__`(*self*: rti.connexdds.PublisherQos, *other*: rti.connexdds.PublisherQos) -> None

Create a copy of a PublisherQos object.

`__lshift__` (**args*, ***kwargs*)

Overloaded function.

1. `__lshift__`(*self*: rti.connexdds.PublisherQos, *arg0*: rti.connexdds.Presentation) -> rti.connexdds.PublisherQos

Set the PresentationQoS.

2. `__lshift__`(*self*: rti.connexdds.PublisherQos, *arg0*: rti.connexdds.Partition) -> rti.connexdds.PublisherQos

Set the PartitionQoS.

3. `__lshift__`(*self*: rti.connexdds.PublisherQos, *arg0*: rti.connexdds.GroupData) -> rti.connexdds.PublisherQos

Set the GroupDataQoS.

4. `__lshift__`(*self*: rti.connexdds.PublisherQos, *arg0*: rti.connexdds.EntityFactory) -> rti.connexdds.PublisherQos

Set the EntityFactoryQoS.

5. `__lshift__`(*self*: rti.connexdds.PublisherQos, *arg0*: rti.connexdds.AsynchronousPublisher) -> rti.connexdds.PublisherQos

Set the AsynchronousPublisherQoS.

6. `__lshift__`(*self*: rti.connexdds.PublisherQos, *arg0*: rti.connexdds.ExclusiveArea) -> rti.connexdds.PublisherQos

Set the ExclusiveAreaQoS.

7. `__lshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.EntityName) -> rti.connextdds.PublisherQos`

Set the EntityNameQoS.

`__module__` = `'rti.connextdds'`

`__ne__` (*self*: rti.connextdds.PublisherQos, *arg0*: rti.connextdds.PublisherQos) → bool

Test for inequality.

`__rshift__` (**args*, ***kwargs*)

Overloaded function.

1. `__rshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.Presentation) -> rti.connextdds.PublisherQos`

Get the PresentationQoS.

2. `__rshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.Partition) -> rti.connextdds.PublisherQos`

Get the PartitionQoS.

3. `__rshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.GroupData) -> rti.connextdds.PublisherQos`

Get the GroupDataQoS.

4. `__rshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.EntityFactory) -> rti.connextdds.PublisherQos`

Get the EntityFactoryQoS.

5. `__rshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.AsynchronousPublisher) -> rti.connextdds.PublisherQos`

Get the AsynchronousPublisherQoS.

6. `__rshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.ExclusiveArea) -> rti.connextdds.PublisherQos`

Get the ExclusiveAreaQoS.

7. `__rshift__(self: rti.connextdds.PublisherQos, arg0: rti.connextdds.EntityName) -> rti.connextdds.PublisherQos`

Get the EntityNameQoS.

`__str__` (*self*: rti.connextdds.PublisherQos) → str

property asynchronous_publisher

Get/set AsynchronousPublisher QoS.

property entity_factory

Get/set EntityFactory QoS.

property entity_name

Get/set EntityName QoS.

property exclusive_area

Get/set ExclusiveArea QoS.

property group_data

Get/set GroupData QoS.

property partition

Get/set Partition QoS.

property presentation

Get/set Presentation QoS.

to_string (*self*: rti.connexdds.PublisherQos, *format*: rti.connexdds.QosPrintFormat = *QosPrintFormat()*, *base*: *Optional*[rti.connexdds.PublisherQos] = *None*, *print_all*: *bool* = *False*) → str

Convert QoS to string based on params.

class rti.connexdds.**PublisherSeq**

Bases: pybind11_object

__add__ (*self*: rti.connexdds.PublisherSeq, *arg0*: rti.connexdds.PublisherSeq) → *rti.connexdds.PublisherSeq*

__bool__ (*self*: rti.connexdds.PublisherSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.PublisherSeq, *x*: rti.connexdds.Publisher) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.PublisherSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.PublisherSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.PublisherSeq, *arg0*: rti.connexdds.PublisherSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.PublisherSeq, *s*: slice) -> rti.connexdds.PublisherSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.PublisherSeq, *arg0*: int) -> rti.connexdds.Publisher

__hash__ = None

__iadd__ (*self*: rti.connexdds.PublisherSeq, *arg0*: rti.connexdds.PublisherSeq) → *rti.connexdds.PublisherSeq*

__imul__ (*self*: rti.connexdds.PublisherSeq, *arg0*: int) → *rti.connexdds.PublisherSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.PublisherSeq) -> None
2. **__init__**(*self*: rti.connexdds.PublisherSeq, *arg0*: rti.connexdds.PublisherSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.PublisherSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.PublisherSeq) → Iterator

__len__ (*self*: rti.connexdds.PublisherSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.PublisherSeq, *arg0*: int) → *rti.connexdds.PublisherSeq*

__ne__ (*self*: rti.connexdds.PublisherSeq, *arg0*: rti.connexdds.PublisherSeq) → bool

__rmul__ (*self*: rti.connexdds.PublisherSeq, *arg0*: int) → *rti.connexdds.PublisherSeq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.PublisherSeq, *arg0*: int, *arg1*: rti.connexdds.Publisher) -> None
2. **__setitem__**(*self*: rti.connexdds.PublisherSeq, *arg0*: slice, *arg1*: rti.connexdds.PublisherSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.PublisherSeq, *x*: rti.connexdds.Publisher) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.PublisherSeq) → None

Clear the contents

count (*self*: rti.connexdds.PublisherSeq, *x*: rti.connexdds.Publisher) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.PublisherSeq, *L*: rti.connexdds.PublisherSeq) -> None

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.PublisherSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.PublisherSeq, *i*: int, *x*: rti.connextdds.Publisher) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connextdds.PublisherSeq) -> rti.connextdds.Publisher`

Remove and return the last item

2. `pop(self: rti.connextdds.PublisherSeq, i: int) -> rti.connextdds.Publisher`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.PublisherSeq, *x*: rti.connextdds.Publisher) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connextdds.QosPolicyCount

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.QosPolicyCount, *arg0*: rti.connextdds.QosPolicyCount) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.QosPolicyCount, *arg0*: rti.connextdds.QosPolicyCount) → bool

Test for inequality.

property count

The counter.

property policy

The policy class.

class rti.connextdds.QosPolicyCountSeq

Bases: pybind11_object

__add__ (*self*: rti.connextdds.QosPolicyCountSeq, *arg0*: rti.connextdds.QosPolicyCountSeq) → rti.connextdds.QosPolicyCountSeq

__bool__ (*self*: rti.connextdds.QosPolicyCountSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connextdds.QosPolicyCountSeq, *x*: rti.connextdds.QosPolicyCount) → bool

Return true the container contains *x*

`__delitem__` (*args, **kwargs)

Overloaded function.

1. `__delitem__(self: rti.connexdds.QosPolicyCountSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connexdds.QosPolicyCountSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (self: rti.connexdds.QosPolicyCountSeq, arg0: rti.connexdds.QosPolicyCountSeq) → bool

`__getitem__` (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connexdds.QosPolicyCountSeq, s: slice) -> rti.connexdds.QosPolicyCountSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.QosPolicyCountSeq, arg0: int) -> rti.connexdds.QosPolicyCount`

`__hash__` = None

`__iadd__` (self: rti.connexdds.QosPolicyCountSeq, arg0: rti.connexdds.QosPolicyCountSeq) → *rti.connexdds.QosPolicyCountSeq*

`__imul__` (self: rti.connexdds.QosPolicyCountSeq, arg0: int) → *rti.connexdds.QosPolicyCountSeq*

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.QosPolicyCountSeq) -> None`
2. `__init__(self: rti.connexdds.QosPolicyCountSeq, arg0: rti.connexdds.QosPolicyCountSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.QosPolicyCountSeq, arg0: Iterable) -> None`

`__iter__` (self: rti.connexdds.QosPolicyCountSeq) → Iterator

`__len__` (self: rti.connexdds.QosPolicyCountSeq) → int

`__module__` = 'rti.connexdds'

`__mul__` (self: rti.connexdds.QosPolicyCountSeq, arg0: int) → *rti.connexdds.QosPolicyCountSeq*

`__ne__` (self: rti.connexdds.QosPolicyCountSeq, arg0: rti.connexdds.QosPolicyCountSeq) → bool

`__rmul__` (self: rti.connexdds.QosPolicyCountSeq, arg0: int) → *rti.connexdds.QosPolicyCountSeq*

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connexdds.QosPolicyCountSeq, arg0: int, arg1: rti.connexdds.QosPolicyCount) -> None`
2. `__setitem__(self: rti.connexdds.QosPolicyCountSeq, arg0: slice, arg1: rti.connexdds.QosPolicyCountSeq) -> None`

Assign list elements using a slice object

append (self: rti.connexdds.QosPolicyCountSeq, x: rti.connexdds.QosPolicyCount) → None

Add an item to the end of the list

clear (self: rti.connexdds.QosPolicyCountSeq) → None

Clear the contents

count (self: rti.connexdds.QosPolicyCountSeq, x: rti.connexdds.QosPolicyCount) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connexdds.QosPolicyCountSeq, L: rti.connexdds.QosPolicyCountSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.QosPolicyCountSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connexdds.QosPolicyCountSeq, i: int, x: rti.connexdds.QosPolicyCount) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connexdds.QosPolicyCountSeq) -> rti.connexdds.QosPolicyCount`

Remove and return the last item

2. `pop(self: rti.connexdds.QosPolicyCountSeq, i: int) -> rti.connexdds.QosPolicyCount`

Remove and return the item at index i

remove (self: rti.connexdds.QosPolicyCountSeq, x: rti.connexdds.QosPolicyCount) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class rti.connexdds.QosPrintFormat

Bases: pybind11_object

__eq__ (self: rti.connexdds.QosPrintFormat, arg0: rti.connexdds.QosPrintFormat) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connextdds.QosPrintFormat) -> None

Initializes the properties with the default values.

2. **__init__**(self: rti.connextdds.QosPrintFormat, indent: int, private: bool, is_standalone: bool) -> None

Initializes the properties.

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.QosPrintFormat, arg0: rti.connextdds.QosPrintFormat) → bool

Test for inequality.

property indent

The value of indent.

property is_standalone

Print XML preamble toggle.

property print_private

Print private QoS field toggle.

class rti.connextdds.QosProvider

Bases: pybind11_object

__eq__ (self: rti.connextdds.QosProvider, arg0: rti.connextdds.QosProvider) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connextdds.QosProvider, uri: str, profile: str) -> None

Creates a QosProvider fetching QoS configuration from the specified URI.

2. **__init__**(self: rti.connextdds.QosProvider, uri: str) -> None

Create a QosProvider fetching QoS configuration from the specified URI.

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.QosProvider, arg0: rti.connextdds.QosProvider) → bool

Test for inequality.

create_participant_from_config (*args, **kwargs)

Overloaded function.

1. `create_participant_from_config(self: rti.connextdds.QosProvider, config: str) -> rti.connextdds.DomainParticipant`

Create a DomainParticipant given a configuration name from a description provided in an XML configuration file that has been loaded by this QosProvider with default parameters.

2. `create_participant_from_config(self: rti.connextdds.QosProvider, config: str, params: rti.connextdds.DomainParticipantConfigParams) -> rti.connextdds.DomainParticipant`

Create a DomainParticipant given a configuration name from a description provided in an XML configuration file that has been loaded by this QosProvider.

property datareader_qos

Get a copy of the DataReaderQos currently associated with the QosProvider.

datareader_qos_from_profile (*self: rti.connextdds.QosProvider, profile: str*) → *rti.connextdds.DataReaderQos*

Get the DataReaderQos from a qos profile.

property datawriter_qos

Get a copy of the DataWriterQos currently associated with the QosProvider.

datawriter_qos_from_profile (*self: rti.connextdds.QosProvider, profile: str*) → *rti.connextdds.DataWriterQos*

Get the DataWriterQos from a qos profile.

default = <rti.connextdds.QosProvider object>

property default_library

The default library associated with this QosProvider (None if not set).

property default_profile

The default profile associated with this QosProvider (None if not set).

property default_profile_library

The library of the default profile associated with this QosProvider (None if not set).

default_provider_params = <rti.connextdds.QosProviderParams object>

get_topic_datareader_qos (*self: rti.connextdds.QosProvider, topic_name: str*) → *rti.connextdds.DataReaderQos*

Get the DataReaderQos associated with a given Topic name.

get_topic_datawriter_qos (*self: rti.connextdds.QosProvider, topic_name: str*) → *rti.connextdds.DataWriterQos*

Get the DataWriterQos associated with a given Topic name.

get_topic_name_qos (*self: rti.connextdds.QosProvider, topic_name: str*) → *rti.connextdds.TopicQos*

Get the TopicQos associated with a given Topic name.

load_profiles (*self*: rti.connexdds.QosProvider) → None

Load the XML QoS profiles from this QosProvider.

property participant_qos

Get a copy of the DomainParticipantQos currently associated with the QosProvider.

participant_qos_from_profile (*self*: rti.connexdds.QosProvider, *profile_name*: str) → *rti.connexdds.DomainParticipantQos*

Get the DomainParticipantQos from a qos profile.

property profiles_loaded

Check if the profiles from this QosProvider have been loaded.

property provider_params

Get a copy of or set the QosProviderParams for this QosProvider.

property publisher_qos

Get a copy of the PublisherQos currently associated with the QosProvider.

publisher_qos_from_profile (*self*: rti.connexdds.QosProvider, *profile*: str) → *rti.connexdds.PublisherQos*

Get the PublisherQos from a qos profile.

property qos_profile_libraries

Get a list of the QoS profile libraries loaded by the QosProvider.

qos_profiles (*self*: rti.connexdds.QosProvider, *library*: str) → *rti.connexdds.StringSeq*

Get a list of the QoS profiles loaded the specified library of the QosProvider.

reload_profiles (*self*: rti.connexdds.QosProvider) → None

Reload the XML QoS profiles from this QosProvider.

static reset_default () → None

Reset the settings of the default QosProvider.

set_topic_datareader_qos (*self*: rti.connexdds.QosProvider, *profile_name*: str, *topic_name*: str) → *rti.connexdds.DataReaderQos*

Set the DataReaderQos for a given Topic name.

set_topic_datawriter_qos (*self*: rti.connexdds.QosProvider, *profile_name*: str, *topic_name*: str) → *rti.connexdds.DataWriterQos*

Set the DataWriterQos for a given Topic name.

set_topic_name_qos (*self*: rti.connexdds.QosProvider, *profile_name*: str, *topic_name*: str) → *rti.connexdds.TopicQos*

Set the TopicQos for a given Topic name.

property subscriber_qos

Get a copy of the SubscriberQos currently associated with this QosProvider.

subscriber_qos_from_profile (*self*: rti.connexdds.QosProvider, *profile*: str) →
rti.connexdds.SubscriberQos

Get the SubscriberQos from a qos profile.

property topic_qos

Get a copy of the TopicQos currently associated with the QosProvider.

topic_qos_from_profile (*self*: rti.connexdds.QosProvider, *profile_name*: str) →
rti.connexdds.TopicQos

Get the TopicQos from a qos profile.

type (*args, **kwargs)

Overloaded function.

1. type(*self*: rti.connexdds.QosProvider, *library*: str, *name*: str) -> object

Get a DynamicType from a type library in the QosProvider.

2. type(*self*: rti.connexdds.QosProvider, *name*: str) -> object

Get a DynamicType from the QosProvider.

property type_libraries

Get a list of the type libraries loaded by this QosProvider.

unload_profiles (*self*: rti.connexdds.QosProvider) → None

Unload the XML QoS profiles from this QosProvider.

class rti.connexdds.QosProviderParams

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.QosProviderParams, *arg0*: rti.connexdds.QosProviderParams) → bool
 Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.QosProviderParams) → None
 Create a QosProviderParams with default settings.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.QosProviderParams, *arg0*: rti.connexdds.QosProviderParams) → bool
 Test for inequality.

property ignore_environment_profile

Choose whether or not to ignore the NDDS_QOS_PROFILES.

property ignore_resource_profile

Choose whether or not to ignore NDDS_QOS_PROFILES.xml.

property ignore_user_profile

Choose whether or not to ignore USER_QOS_PROFILES.xml.

property string_profile

Sequence of strings containing a XML document to load.

property url_profile

Sequence of XML documents to load.

class `rti.connexdds.Query`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.Query`, *arg0*: `rti.connexdds.Query`) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: `rti.connexdds.Query`, *reader*: `rti.connexdds.IDataReader`, *expression*: str) -> None

Init a Query for a DataReader with an expression.

2. **__init__**(*self*: `rti.connexdds.Query`, *reader*: `rti.connexdds.IDataReader`, *expression*: str, *params*: `rti.connexdds.StringSeq`) -> None

Init a Query for a DataReader with an expression and parameters.

__iter__ (*self*: `rti.connexdds.Query`) → Iterator

__len__ (*self*: `rti.connexdds.Query`) → int

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.Query`, *arg0*: `rti.connexdds.Query`) → bool

Test for inequality.

add_parameter (*self*: `rti.connexdds.Query`, *arg0*: str) → None

Appends a parameter.

property data_reader

The DataReader as an AnyDataReader.

property expression

The expression.

property name

The filter name.

property parameters

The parameters for the expression.

property parameters_length

The parameter sequence length.


```
class rti.connexdds.QueryCondition
```

```
Bases: IReadCondition
```

```
__eq__ (self: rti.connexdds.QueryCondition, arg0: rti.connexdds.QueryCondition) → bool
```

Test for equality.

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connexdds.QueryCondition, query: rti.connexdds.Query, status: rti.connexdds.DataState) -> None`

Create a QueryCondition.

2. `__init__(self: rti.connexdds.QueryCondition, query: rti.connexdds.Query, status: rti.connexdds.DataState, handler: Callable[[rti.connexdds.QueryCondition], None]) -> None`

Create a QueryCondition.

3. `__init__(self: rti.connexdds.QueryCondition, condition: rti.connexdds.ICondition) -> None`

Cast a condition to a QueryCondition.

```
__len__ (self: rti.connexdds.QueryCondition) → int
```

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.QueryCondition, arg0: rti.connexdds.QueryCondition) → bool
```

Test for inequality.

```
property expression
```

The expression.

```
property parameters
```

The parameters for the expression.

```
property parameters_length
```

The parameter sequence length.

```
reset_handler (self: rti.connexdds.QueryCondition) → None
```

Resets the handler for this QueryCondition.

```
set_handler (self: rti.connexdds.QueryCondition, func: Callable[[rti.connexdds.QueryCondition], None]) → None
```

Set a handler function for this QueryCondition.

```
set_handler_no_args (self: rti.connexdds.QueryCondition, func: Callable[[], None]) → None
```

Set a handler function receiving no arguments.

```
class rti.connexdds.Rank
```

```
Bases: pybind11_object
```

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connextdds.Rank) -> None`

Create a default Rank object.

2. `__init__(self: rti.connextdds.Rank, sample_rank: int, generation_rank: int, absolute_generation_rank: int) -> None`

Create a GenerationCount object with the provided `disposed_count` and `no_writers` count.

```
__module__ = 'rti.connextdds'
```

```
property absolute_generation
```

Get the absolute generation rank of the sample.

```
property generation
```

Get the generation rank of the sample.

```
property sample
```

Get the sample rank of the sample.

```
class rti.connextdds.ReadCondition
```

Bases: *IReadCondition*

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connextdds.ReadCondition, reader: rti.connextdds.IAnyDataReader, status: rti.connextdds.DataState) -> None`

Create a ReadCondition.

2. `__init__(self: rti.connextdds.ReadCondition, reader: rti.connextdds.IAnyDataReader, status: rti.connextdds.DataState, handler: Callable[[rti.connextdds.ReadCondition], None]) -> None`

Create a ReadCondition.

3. `__init__(self: rti.connextdds.ReadCondition, arg0: rti.connextdds.ICondition) -> None`

Cast a compatible Condition to a ReadCondition.

```
__module__ = 'rti.connextdds'
```

```
reset_handler (self: rti.connextdds.ReadCondition) → None
```

Resets the handler for this ReadCondition.

```
set_handler_no_args (self: rti.connextdds.ReadCondition, func: Callable[[], None]) → None
```

Set a handler function receiving no arguments.

```
class rti.connextdds.ReaderDataLifecycle
```

Bases: `pybind11_object`

__eq__ (*self*: rti.connexdds.ReaderDataLifecycle, *arg0*: rti.connexdds.ReaderDataLifecycle) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ReaderDataLifecycle) -> None

Creates the default policy.

2. **__init__**(*self*: rti.connexdds.ReaderDataLifecycle, *nowriter_delay*: rti.connexdds.Duration, *disposed_samples_delay*: rti.connexdds.Duration) -> None

Creates an instance with the specified nowriter and disposed-samples purge delays.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ReaderDataLifecycle, *arg0*: rti.connexdds.ReaderDataLifecycle) → bool

Test for inequality.

static auto_purge_disposed_samples (*delay*: rti.connexdds.Duration) → *rti.connexdds.ReaderDataLifecycle*

Returns a policy where only the disposed-samples purge delay is enabled with a specified duration.

static auto_purge_no_writer_samples (*delay*: rti.connexdds.Duration) → *rti.connexdds.ReaderDataLifecycle*

Returns a policy where only the no-writer purge delay is enabled with a specified duration.

property autopurge_disposed_instances_delay

Minimum duration for which the DataReader will maintain an instance once its *instance_state* becomes *InstanceStateKind.NOT_ALIVE_DISPOSED*.

This property's getter returns a deep copy.

property autopurge_disposed_samples_delay

Minimum duration for which the DataReader will maintain information regarding an instance once its *instance_state* becomes *InstanceStateKind.NOT_ALIVE_DISPOSED*.

This property's getter returns a deep copy.

property autopurge_nowriter_instances_delay

Minimum duration for which the DDSDataReader will maintain information about a received instance once its *instance_state* becomes *DDS_NOT_ALIVE_NO_WRITERS_INSTANCE_STATE* and there are no samples for the instance in the DataReader queue.

property autopurge_nowriter_samples_delay

Minimum duration for which the DataReader will maintain information regarding an instance once its *instance_state* becomes *InstanceStateKind.NOT_ALIVE_NO_WRITERS*.

This property's getter returns a deep copy.

`no_auto_purge` = <rti.connexdds.ReaderDataLifecycle object>

class rti.connexdds.ReceiverPool

Bases: pybind11_object

`__eq__` (*self*: rti.connexdds.ReceiverPool, *arg0*: rti.connexdds.ReceiverPool) → bool

Test for equality.

`__hash__` = None

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.ReceiverPool) -> None`

Creates the default policy.

2. `__init__(self: rti.connexdds.ReceiverPool, thread: rti.connexdds.ThreadSettings, buffer_size: int, buffer_alignment: int) -> None`

Creates an instance with the thread settings, buffer size and buffer alignment configuration.

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.ReceiverPool, *arg0*: rti.connexdds.ReceiverPool) → bool

Test for inequality.

property `buffer_alignment`

The receive buffer alignment.

property `buffer_size`

The length of the buffer used to store the incoming raw data.

property `thread`

Configures the receiver pool thread(s).

class rti.connexdds.Reliability

Bases: pybind11_object

`__eq__` (*self*: rti.connexdds.Reliability, *arg0*: rti.connexdds.Reliability) → bool

Test for equality.

`__hash__` = None

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.Reliability) -> None`

Creates a best-effort policy.

2. `__init__(self: rti.connexdds.Reliability, kind: rti.connexdds.ReliabilityKind, max_blocking_time: rti.connexdds.Duration = Duration.from_milliseconds(100)) -> None`

Creates an instance with the specified reliability kind and optionally a specific maximum blocking time.

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.Reliability, arg0: rti.connexdds.Reliability) → bool
```

Test for inequality.

```
property acknowledgment_kind
```

The kind of reliable acknowledgment.

```
best_effort = <rti.connexdds.Reliability object>
```

```
property instance_state_consistency_kind
```

Whether instance state consistency is enabled.

```
property kind
```

The reliability kind.

```
property max_blocking_time
```

The maximum time a DataWriter may block on a call to write().

This property's getter returns a deep copy.

```
static reliable (max_blocking_time: rti.connexdds.Duration =  
                  Duration.from_milliseconds(100)) → rti.connexdds.Reliability
```

Creates a policy with ReliabilityKind.RELIABLE and optionally a max blocking time.

```
class rti.connexdds.ReliabilityKind
```

Bases: pybind11_object

```
BEST_EFFORT = <ReliabilityKind.BEST_EFFORT: 0>
```

```
RELIABLE = <ReliabilityKind.RELIABLE: 1>
```

```
class ReliabilityKind
```

Bases: pybind11_object

Members:

BEST_EFFORT : Indicates that it is acceptable to not retry propagation of any samples.

Presumably new values for the samples are generated often enough that it is not necessary to re-send or acknowledge any samples.

[default] for DataReader and Topic.

RELIABLE : Specifies that RTI Connex will attempt to deliver all samples in its history. Missed samples may be retried.

In steady-state (no modifications communicated via the DataWriter), RTI Connex guarantees that all samples in the DataWriter history will eventually be delivered to all the DataReader objects (subject to timeouts that indicate loss of communication with a particular Subscriber).

Outside steady state, the HISTORY and RESOURCE_LIMITS policies will determine how samples become part of the history and whether samples can be discarded from it.

[default] for DataWriter.

BEST_EFFORT = <ReliabilityKind.BEST_EFFORT: 0>

RELIABLE = <ReliabilityKind.RELIABLE: 1>

__eq__ (*self*: object, *other*: object) → bool

__getstate__ (*self*: object) → int

__hash__ (*self*: object) → int

__index__ (*self*: rti.connexdds.ReliabilityKind.ReliabilityKind) → int

__init__ (*self*: rti.connexdds.ReliabilityKind.ReliabilityKind, *value*: int) → None

__int__ (*self*: rti.connexdds.ReliabilityKind.ReliabilityKind) → int

__members__ = {'BEST_EFFORT': <ReliabilityKind.BEST_EFFORT: 0>, 'RELIABLE': <ReliabilityKind.RELIABLE: 1>}

__module__ = 'rti.connexdds'

__ne__ (*self*: object, *other*: object) → bool

__repr__ (*self*: object) → str

__setstate__ (*self*: rti.connexdds.ReliabilityKind.ReliabilityKind, *state*: int) → None

__str__ ()

name(*self*: handle) -> str

property name

property value

__eq__ (*self*: rti.connexdds.ReliabilityKind, *arg0*: rti.connexdds.ReliabilityKind) → bool

Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.ReliabilityKind, *arg0*: rti.connexdds.ReliabilityKind) → bool

Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.ReliabilityKind, *arg0*: rti.connexdds.ReliabilityKind) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ReliabilityKind) -> None

Initializes enum to 0.

2. `__init__(self: rti.connextdds.ReliabilityKind, arg0: rti.connextdds.ReliabilityKind.ReliabilityKind) -> None`

Copy constructor.

`__int__(self: rti.connextdds.ReliabilityKind) -> rti.connextdds.ReliabilityKind.ReliabilityKind`

Int conversion.

`__le__(self: rti.connextdds.ReliabilityKind, arg0: rti.connextdds.ReliabilityKind) -> bool`

Apply operator to underlying enumerated values.

`__lt__(self: rti.connextdds.ReliabilityKind, arg0: rti.connextdds.ReliabilityKind) -> bool`

Apply operator to underlying enumerated values.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.ReliabilityKind, arg0: rti.connextdds.ReliabilityKind) -> bool`

Apply operator to underlying enumerated values.

`__str__(self: rti.connextdds.ReliabilityKind) -> str`

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connextdds.ReliableReaderActivityChangedStatus

Bases: `pybind11_object`

`__init__(*args, **kwargs)`

`__module__ = 'rti.connextdds'`

property active_count

The current number of reliable DataReaders currently matched with this reliable DataWriter.

property inactive_count

The number of reliable DataReaders that have been dropped by this reliable DataWriter because they failed to send acknowledgments in a timely fashion.

property last_instance_handle

The instance handle of the last reliable remote DataReader to be determined inactive.

class rti.connextdds.ReliableWriterCacheChangedStatus

Bases: `pybind11_object`

`__init__(*args, **kwargs)`

`__module__ = 'rti.connextdds'`

property empty_reliable_writer_cache

The number of times the reliable DataWriters's cache of unacknowledged samples has become empty.

property full_reliable_writer_cache

The number of times the reliable DataWriters's cache (or send window) of unacknowledged samples has become full.

property high_watermark_reliable_writer_cache

The number of times the reliable DataWriter's cache of unacknowledged samples has risen to the high watermark.

property low_watermark_reliable_writer_cache

The number of times the reliable DataWriter's cache of unacknowledged samples has fallen to the low watermark.

property replaced_unacknowledged_sample_count

The number of unacknowledged samples that have been replaced in the writer's cache.

property unacknowledged_sample_count

The current number of unacknowledged samples in the DataWriter's cache.

property unacknowledged_sample_count_peak

The highest value that unacknowledged_sample_count has reached until now.

class rti.connextdds.RemoteParticipantPurgeKind

Bases: pybind11_object

LIVELINESS_BASED = <RemoteParticipantPurgeKind.LIVELINESS_BASED: 0>

NO_PURGE = <RemoteParticipantPurgeKind.NO_PURGE: 1>

class RemoteParticipantPurgeKind

Bases: pybind11_object

Members:

LIVELINESS_BASED : [default] Maintain knowledge of the remote participant for as long as it maintains its liveliness contract.

A participant will continue attempting communication with its peers, even if discovery communication with them is lost, as long as the remote participants maintain their liveliness. If both discovery communication and participant liveliness are lost, however, the local participant will remove all records of the remote participant and its contained endpoints, and no further data communication with them will occur until and unless they are rediscovered.

The liveliness contract a participant promises to its peers (its "liveliness lease duration") is specified in its `DiscoveryConfig.participant_liveliness_lease_duration` QoS field. It maintains that contract by writing data to those other participants with a writer that has a `LivelinessKind` of `LivelinessKind.AUTOMATIC` or `LivelinessKind.MANUAL_BY_PARTICIPANT` and by asserting itself (at the `DiscoveryConfig.participant_liveliness_assert_period`) over the Simple Discovery Protocol.

NO_PURGE : Never “forget” a remote participant with which discovery communication has been lost.

If a participant with this behavior loses discovery communication with a remote participant, it will nevertheless remember that remote participant and its endpoints and continue attempting to communicate with them indefinitely.

This value has consequences for a participant’s resource usage. If discovery communication with a remote participant is lost, but the same participant is later rediscovered, any relevant records that remain in the database will be reused. However, if it is not rediscovered, the records will continue to take up space in the database for as long as the local participant remains in existence.

```

LIVELINESS_BASED =
<RemoteParticipantPurgeKind.LIVELINESS_BASED: 0>

NO_PURGE = <RemoteParticipantPurgeKind.NO_PURGE: 1>

__eq__(self: object, other: object) → bool

__getstate__(self: object) → int

__hash__(self: object) → int

__index__(self: rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind)
    → int

__init__(self: rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind,
    value: int) → None

__int__(self: rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind) →
    int

__members__ = {'LIVELINESS_BASED':
<RemoteParticipantPurgeKind.LIVELINESS_BASED: 0>, 'NO_PURGE':
<RemoteParticipantPurgeKind.NO_PURGE: 1>}

__module__ = 'rti.connexdds'

__ne__(self: object, other: object) → bool

__repr__(self: object) → str

__setstate__(self:
    rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind,
    state: int) → None

__str__()
    name(self: handle) -> str

property name

property value

```

__eq__ (*self*: rti.connexdds.RemoteParticipantPurgeKind, *arg0*: rti.connexdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.RemoteParticipantPurgeKind, *arg0*: rti.connexdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.RemoteParticipantPurgeKind, *arg0*: rti.connexdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.RemoteParticipantPurgeKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.RemoteParticipantPurgeKind, *arg0*: rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.RemoteParticipantPurgeKind) → *rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind*

Int conversion.

__le__ (*self*: rti.connexdds.RemoteParticipantPurgeKind, *arg0*: rti.connexdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.RemoteParticipantPurgeKind, *arg0*: rti.connexdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.RemoteParticipantPurgeKind, *arg0*: rti.connexdds.RemoteParticipantPurgeKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.RemoteParticipantPurgeKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.**RequestedDeadlineMissedStatus**

Bases: pybind11_object

`__init__` (*args, **kwargs)

`__module__` = 'rti.connextdds'

property last_instance_handle

Handle to the last instance in the DataReader for which a missed deadline was detected.

property total_count

Total count of the missed deadlines detected for any instance read by the DataReader.

property total_count_change

The delta in missed deadlines detected since the last time the listener was called or the status was read.

class rti.connextdds.RequestedIncompatibleQoSStatus

Bases: pybind11_object

`__init__` (*args, **kwargs)

`__module__` = 'rti.connextdds'

property last_policy

The policy class of one of the policies that was found to be incompatible the last time an incompatibility was detected.

property policies

A list containing, for each policy, the total number of times that the concerned DataReader discovered a DataWriter for the same Topic with an offered QoS that is incompatible with that requested by the DataReader.

total_count (self: rti.connextdds.RequestedIncompatibleQoSStatus) → int

Total count of how many times the concerned DataReader discovered a DataWriter for the same Topic with an offered QoS that is incompatible with that requested by the DataReader.

property total_count_change

The delta in total_count since the last time the listener was called or the status was read.

class rti.connextdds.ResourceLimits

Bases: pybind11_object

`__eq__` (self: rti.connextdds.ResourceLimits, arg0: rti.connextdds.ResourceLimits) → bool

Test for equality.

`__hash__` = None

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.ResourceLimits) -> None`

Creates the default policy.

2. `__init__(self: rti.connextdds.ResourceLimits, max_samples: int, max_instances: int, max_samples_per_instance: int) -> None`

Creates the default policy.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.ResourceLimits, *arg0*: rti.connextdds.ResourceLimits) → bool

Test for inequality.

property initial_instances

The number of instances that a DataWriter or a DataWriter will preallocate.

property initial_samples

The maximum number of data samples per instance that a DataWriter or a DataReader can manage.

property instance_hash_buckets

The number of instances that a DataWriter or a DataWriter will preallocate.

property max_instances

Sets the maximum number of instances that a DataWriter or a DataReader can manage.

property max_samples

Sets the maximum number of data samples that a DataWriter or a DataReader can manage across all instances.

property max_samples_per_instance

Sets the maximum number of data samples per instance that a DataWriter or a DataReader can manage.

class rti.connextdds.RtpsReliableReaderProtocol

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.RtpsReliableReaderProtocol, *arg0*: rti.connextdds.RtpsReliableReaderProtocol) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.RtpsReliableReaderProtocol) → None

Create an RtpsReliableReaderProtocol policy with default settings.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.RtpsReliableReaderProtocol, *arg0*: rti.connextdds.RtpsReliableReaderProtocol) → bool

Test for inequality.

property app_ack_period

Get/set the period at which application-level acknowledgment messages are sent.

property heartbeat_suppression_duration

Get/set the duration to ignore consecutive heartbeats from a DataWriter.

property max_heartbeat_response_delay

Get/set the maximum time to respond with an ACKNACK to a DataWriter's heartbeat.

property min_heartbeat_response_delay

Get/set the minimum time to respond with an ACKNACK to a DataWriter's heartbeat.

property nack_period

Get/set the period at which to send NACKs for receiving historical data.

property receive_window_size

Get/set the number of received out-of-order samples a reader can keep at a time.

property round_trip_time

Get/set the estimated duration from sending a NACK to receiving a repair of a sample.

property samples_per_app_ack

Get/set the minimum number of samples acknowledged by one application-level acknowledgment message.

class rti.connextdds.RtpsReliableWriterProtocol

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.RtpsReliableWriterProtocol, *arg0*: rti.connextdds.RtpsReliableWriterProtocol) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.RtpsReliableWriterProtocol) → None

Create a default RtpsReliableWriterProtocol policy.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.RtpsReliableWriterProtocol, *arg0*: rti.connextdds.RtpsReliableWriterProtocol) → bool

Test for inequality.

property disable_positive_acks_decrease_sample_keep_duration_factor

Controls rate of contraction of dynamic sample keep duration.

property disable_positive_acks_enable_adaptive_sample_keep_duration

Enables dynamic adjustment of sample keep duration in response to congestion.

property disable_positive_acks_increase_sample_keep_duration_factor

Controls rate of growth of dynamic sample keep duration.

property disable_positive_acks_max_sample_keep_duration

The maximum duration a sample is queued for ACK-disabled readers.

This property's getter returns a deep copy.

property disable_positive_acks_min_sample_keep_duration

The minimum duration a sample is queued for ACK-disabled readers.

This property's getter returns a deep copy.

property disable_repair_piggyback_heartbeat

Prevents piggyback heartbeats from being sent with repair samples.

property enable_multicast_periodic_heartbeat

Whether periodic heartbeat messages are sent over multicast.

property fast_heartbeat_period

An alternative heartbeat period used when a reliable writer needs to flush its unacknowledged samples more quickly.

This property's getter returns a deep copy.

property heartbeat_period

The period at which to send heartbeats.

This property's getter returns a deep copy.

property heartbeats_per_max_samples

The number of heartbeats per current send window.

property high_watermark

When the number of unacknowledged samples in the current send window of a reliable writer meets or exceeds this threshold, `StatusMask.reliable_writer_cache_changed` is considered to have changed.

property inactivate_nonprogressing_readers

Whether to treat remote readers as inactive when their NACKs do not progress.

property late_joiner_heartbeat_period

An alternative heartbeat period used when a reliable reader joins late and needs to be caught up on cached samples of a reliable writer more quickly than the normal heartbeat rate.

This property's getter returns a deep copy.

property low_watermark

When the number of unacknowledged samples in the current send window of a reliable writer meets or falls below this threshold, the `StatusMask.reliable_writer_cache_changed` is considered to have changed.

property max_bytes_per_nack_response

The maximum total message size when resending dropped samples.

property max_heartbeat_retries

The maximum number of periodic heartbeat retries before marking a remote reader as inactive.

property max_nack_response_delay

The maximum delay to respond to a NACK.

This property's getter returns a deep copy.

property max_send_window_size

Maximum size of send window of unacknowledged samples.

property min_nack_response_delay

The minimum delay to respond to a NACK.

This property's getter returns a deep copy.

property min_send_window_size

Minimum size of send window of unacknowledged samples.

property multicast_resend_threshold

The minimum number of requesting readers needed to trigger a multicast resend.

property nack_suppression_duration

The duration for ignoring consecutive NACKs that may trigger redundant repairs.

This property's getter returns a deep copy.

property samples_per_virtual_heartbeat

The number of samples that a reliable writer has to publish before sending a virtual heartbeat.

property send_window_decrease_factor

Decreases send window size by this percentage when reacting dynamically to network conditions.

property send_window_increase_factor

Increases send window size by this percentage when reacting dynamically to network conditions.

property send_window_update_period

Period in which send window may be dynamically changed.

This property's getter returns a deep copy.

property virtual_heartbeat_period

The period at which to send virtual heartbeats. Virtual heartbeats inform the reliable reader about the range of samples currently present, for each virtual GUID, in the reliable writer's queue.

This property's getter returns a deep copy.

class rti.connexdds.RtpsReservedPortKindMask

Bases: pybind11_object

ALL = 1111

BUILTIN_MULTICAST = 0010

BUILTIN_UNICAST = 0001

DEFAULT_MASK = 0111

NONE = 0000

USER_MULTICAST = 1000

USER_UNICAST = 0100

__and__ (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: rti.connexdds.RtpsReservedPortKindMask) → *rti.connexdds.RtpsReservedPortKindMask*

Bitwise logical AND of masks.

__bool__ (*self*: rti.connexdds.RtpsReservedPortKindMask) → bool

Test if any bits are set.

__contains__ (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: rti.connexdds.RtpsReservedPortKindMask) → bool

__eq__ (**args*, ***kwargs*)

Overloaded function.

1. **__eq__**(*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: rti.connexdds.RtpsReservedPortKindMask) -> bool

Compare masks for equality.

2. **__eq__**(*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: rti.connexdds.RtpsReservedPortKindMask) -> bool

Test for equality.

__getitem__ (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: int) → bool

Get individual mask bit.

__hash__ = None

__iand__ (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: rti.connexdds.RtpsReservedPortKindMask) → *rti.connexdds.RtpsReservedPortKindMask*

Set mask to logical AND with another mask.

__ilshift__ (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: int) → *rti.connexdds.RtpsReservedPortKindMask*

Left shift bits in mask.

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.RtpsReservedPortKindMask) -> None

Create a RtpsReservedPortKindMask equivalent to RtpsReservedPortKindMask.NONE

2. **__init__**(*self*: rti.connexdds.RtpsReservedPortKindMask, *value*: int) -> None

Creates a mask from the bits in an integer.

__int__ (*self*: rti.connexdds.RtpsReservedPortKindMask) → int

Convert mask to int.

`__ior__` (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: rti.connexdds.RtpsReservedPortKindMask) → *rti.connexdds.RtpsReservedPortKindMask*

Set mask to logical OR with another mask.

`__irshift__` (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: int) → *rti.connexdds.RtpsReservedPortKindMask*

Right shift bits in mask.

`__ixor__` (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: rti.connexdds.RtpsReservedPortKindMask) → *rti.connexdds.RtpsReservedPortKindMask*

Set mask to logical XOR with another mask.

`__lshift__` (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: int) → *rti.connexdds.RtpsReservedPortKindMask*

Left shift bits in mask.

`__module__` = 'rti.connexdds'

`__ne__` (**args*, ***kwargs*)

Overloaded function.

1. `__ne__(self: rti.connexdds.RtpsReservedPortKindMask, arg0: rti.connexdds.RtpsReservedPortKindMask) -> bool`

Compare masks for inequality.

2. `__ne__(self: rti.connexdds.RtpsReservedPortKindMask, arg0: rti.connexdds.RtpsReservedPortKindMask) -> bool`

Test for inequality.

`__or__` (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: rti.connexdds.RtpsReservedPortKindMask) → *rti.connexdds.RtpsReservedPortKindMask*

Bitwise logical OR of masks.

`__repr__` (*self*: rti.connexdds.RtpsReservedPortKindMask) → str

Convert mask to string.

`__rshift__` (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: int) → *rti.connexdds.RtpsReservedPortKindMask*

Right shift bits in mask.

`__setitem__` (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: int, *arg1*: bool) → None

Set individual mask bit

`__str__` (*self*: rti.connexdds.RtpsReservedPortKindMask) → str

Convert mask to string.

`__xor__` (*self*: rti.connexdds.RtpsReservedPortKindMask, *arg0*: rti.connexdds.RtpsReservedPortKindMask) → *rti.connexdds.RtpsReservedPortKindMask*

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (*args, **kwargs)

Overloaded function.

1. flip(self: rti.connexdds.RtpsReservedPortKindMask) -> rti.connexdds.RtpsReservedPortKindMask

Flip all bits in the mask.

2. flip(self: rti.connexdds.RtpsReservedPortKindMask, pos: int) -> rti.connexdds.RtpsReservedPortKindMask

Flip the mask bit at the specified position.

reset (*args, **kwargs)

Overloaded function.

1. reset(self: rti.connexdds.RtpsReservedPortKindMask) -> rti.connexdds.RtpsReservedPortKindMask

Clear all bits in the mask.

2. reset(self: rti.connexdds.RtpsReservedPortKindMask, pos: int) -> rti.connexdds.RtpsReservedPortKindMask

Clear the mask bit at the specified position.

set (*args, **kwargs)

Overloaded function.

1. set(self: rti.connexdds.RtpsReservedPortKindMask) -> rti.connexdds.RtpsReservedPortKindMask

Set all bits in the mask.

2. set(self: rti.connexdds.RtpsReservedPortKindMask, pos: int, value: bool = True) -> rti.connexdds.RtpsReservedPortKindMask

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (self: rti.connexdds.RtpsReservedPortKindMask, pos: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (self: rti.connexdds.RtpsReservedPortKindMask) → bool

Test if all bits are set.

test_any (self: rti.connexdds.RtpsReservedPortKindMask) → bool

Test if any bits are set.

test_none (*self*: rti.connextdds.RtpsReservedPortKindMask) → bool

Test if none of the bits are set.

class rti.connextdds.RtpsWellKnownPorts

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.RtpsWellKnownPorts, *arg0*: rti.connextdds.RtpsWellKnownPorts) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.RtpsWellKnownPorts) -> None

Creates an instance that contains the default RTPS well-known ports.

2. **__init__**(*self*: rti.connextdds.RtpsWellKnownPorts, *port_base*: int, *domain_id_gain*: int, *participant_id_gain*: int, *builtin_multicast_port_offset*: int, *builtin_unicast_port_offset*: int, *user_multicast_port_offset*: int, *user_unicast_port_offset*: int) -> None

Creates an instance with the specified ports.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.RtpsWellKnownPorts, *arg0*: rti.connextdds.RtpsWellKnownPorts) → bool

Test for inequality.

backwards_compatible = <rti.connextdds.RtpsWellKnownPorts object>

property builtin_multicast_port_offset

Additional offset for metatraffic multicast port.

property builtin_unicast_port_offset

Additional offset for metatraffic unicast port.

property domain_id_gain

Tunable domain gain parameter.

interoperable = <rti.connextdds.RtpsWellKnownPorts object>

property participant_id_gain

Tunable participant gain parameter.

property port_base

The port base offset.

property user_multicast_port_offset

Additional offset for usertraffic multicast port.

property user_unicast_port_offset

Additional offset for usertraffic unicast port.

class `rti.connexdds.SampleFlag`

Bases: `pybind11_object`

INTERMEDIATE_REPLY_SEQUENCE = `<rti.connexdds.SampleFlag object>`

INTERMEDIATE_TOPIC_QUERY_SAMPLE = `<rti.connexdds.SampleFlag object>`

LAST_SHARED_READER_QUEUE = `<rti.connexdds.SampleFlag object>`

REDELIVERED = `<rti.connexdds.SampleFlag object>`

REPLICATE = `<rti.connexdds.SampleFlag object>`

__and__ (*self*: `rti.connexdds.SampleFlag`, *arg0*: `rti.connexdds.SampleFlag`) → `rti.connexdds.SampleFlag`

Bitwise logical AND of masks.

__bool__ (*self*: `rti.connexdds.SampleFlag`) → `bool`

Test if any bits are set.

__contains__ (*self*: `rti.connexdds.SampleFlag`, *arg0*: `rti.connexdds.SampleFlag`) → `bool`

__eq__ (*self*: `rti.connexdds.SampleFlag`, *arg0*: `rti.connexdds.SampleFlag`) → `bool`

Compare masks for equality.

__getitem__ (*self*: `rti.connexdds.SampleFlag`, *arg0*: `int`) → `bool`

Get individual mask bit.

__hash__ = `None`

__iand__ (*self*: `rti.connexdds.SampleFlag`, *arg0*: `rti.connexdds.SampleFlag`) → `rti.connexdds.SampleFlag`

Set mask to logical AND with another mask.

__ilshift__ (*self*: `rti.connexdds.SampleFlag`, *arg0*: `int`) → `rti.connexdds.SampleFlag`

Left shift bits in mask.

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.SampleFlag) -> None`

Construct an empty `SampleFlag` with no bits set.

2. `__init__(self: rti.connexdds.SampleFlag, value: int) -> None`

Creates a mask from the bits in an integer.

__int__ (*self*: `rti.connexdds.SampleFlag`) → `int`

Convert mask to int.

__ior__ (*self*: rti.connexdds.SampleFlag, *arg0*: rti.connexdds.SampleFlag) → *rti.connexdds.SampleFlag*

Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.SampleFlag, *arg0*: int) → *rti.connexdds.SampleFlag*

Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.SampleFlag, *arg0*: rti.connexdds.SampleFlag) → *rti.connexdds.SampleFlag*

Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.SampleFlag, *arg0*: int) → *rti.connexdds.SampleFlag*

Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.SampleFlag, *arg0*: rti.connexdds.SampleFlag) → bool

Compare masks for inequality.

__or__ (*self*: rti.connexdds.SampleFlag, *arg0*: rti.connexdds.SampleFlag) → *rti.connexdds.SampleFlag*

Bitwise logical OR of masks.

__rshift__ (*self*: rti.connexdds.SampleFlag, *arg0*: int) → *rti.connexdds.SampleFlag*

Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.SampleFlag, *arg0*: int, *arg1*: bool) → None

Set individual mask bit

__str__ (*self*: rti.connexdds.SampleFlag) → str

__xor__ (*self*: rti.connexdds.SampleFlag, *arg0*: rti.connexdds.SampleFlag) → *rti.connexdds.SampleFlag*

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (**args*, ***kwargs*)

Overloaded function.

1. flip(*self*: rti.connexdds.SampleFlag) -> rti.connexdds.SampleFlag

Flip all bits in the mask.

2. flip(*self*: rti.connexdds.SampleFlag, *pos*: int) -> rti.connexdds.SampleFlag

Flip the mask bit at the specified position.

reset (**args*, ***kwargs*)

Overloaded function.

1. reset(*self*: rti.connexdds.SampleFlag) -> rti.connexdds.SampleFlag

Clear all bits in the mask.

2. `reset(self: rti.connextdds.SampleFlag, pos: int) -> rti.connextdds.SampleFlag`

Clear the mask bit at the specified position.

set (**args*, ***kwargs*)

Overloaded function.

1. `set(self: rti.connextdds.SampleFlag) -> rti.connextdds.SampleFlag`

Set all bits in the mask.

2. `set(self: rti.connextdds.SampleFlag, pos: int, value: bool = True) -> rti.connextdds.SampleFlag`

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (*self*: rti.connextdds.SampleFlag, *pos*: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (*self*: rti.connextdds.SampleFlag) → bool

Test if all bits are set.

test_any (*self*: rti.connextdds.SampleFlag) → bool

Test if any bits are set.

test_none (*self*: rti.connextdds.SampleFlag) → bool

Test if none of the bits are set.

class rti.connextdds.SampleIdentity

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.SampleIdentity, *arg0*: rti.connextdds.SampleIdentity) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.SampleIdentity) -> None`

Create a default SampleIdentity object.

2. `__init__(self: rti.connextdds.SampleIdentity, writer_guid: rti.connextdds.Guid, sequence_number: rti.connextdds.SequenceNumber) -> None`

Create a SampleIdentity with the provided Guid and SequenceNumber.

__module__ = 'rti.connextdds'

`__ne__` (*self*: rti.connexdds.SampleIdentity, *arg0*: rti.connexdds.SampleIdentity) → bool

Test for inequality.

`__str__` (*self*: rti.connexdds.SequenceNumber) → str

`automatic` = <rti.connexdds.SampleIdentity object>

property sequence_number

Get the monotonically increasing 64-bit SequenceNumber that identifies the sample in the data source.

`unknown` = <rti.connexdds.SampleIdentity object>

property writer_guid

Get the 16-byte identifier identifying the virtual Guid.

class rti.connexdds.SampleInfo

Bases: pybind11_object

`__eq__` (*self*: rti.connexdds.SampleInfo, *arg0*: rti.connexdds.SampleInfo) → bool

Check if two SampleInfo are equal.

`__hash__` = None

`__init__` (*args, **kwargs)

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.SampleInfo, *arg0*: rti.connexdds.SampleInfo) → bool

Check if two SampleInfo are not equal.

`__repr__` (*self*: rti.connexdds.SampleInfo) → str

property coherent_set_info

TWhen set, this field provides the information about the coherent set associated with the sample.

property encapsulation_id

The encapsulation kind.

property flag

Flags associated with the sample.

property generation_count

The GenerationCount of the sample.

property instance_handle

Identifies locally the corresponding instance.

property original_publication_virtual_guid

Original publication virtual GUID.

If the Publisher's access_scope is GROUP, this field contains the Publisher virtual GUID that uniquely identifies the DataWriter group.

property original_publication_virtual_sample_identity

Retrieves the information provided by original_publication_virtual_guid and original_publication_virtual_sequence_number combined in a SampleIdentity instance.

property original_publication_virtual_sequence_number

Original publication virtual sequence number.

If the Publisher's access_scope is GROUP, this field contains the Publisher virtual sequence number that uniquely identifies a DDS sample within the DataWriter group.

property publication_handle

Identifies locally the DataWriter that modified the instance.

property publication_sequence_number

Publication sequence number assigned when the DDS sample was written by the DataWriter.

property rank

Get the Rank of the sample.

property reception_sequence_number

Reception sequence number assigned when the DDS sample was committed by the DataReader.

property reception_timestamp

The timestamp when the sample was committed by a DataReader.

property related_original_publication_virtual_guid

The original publication virtual GUID of a related sample.

property related_original_publication_virtual_sample_identity

Retrieves the information provided by related_original_publication_virtual_guid and related_original_publication_virtual_sequence_number combined in a SampleIdentity instance.

property related_original_publication_virtual_sequence_number

The original publication virtual sequence number of a related sample.

property related_source_guid

The application logical data source that is related to the sample.

property related_subscription_guid

The related_reader_guid associated with the sample.

property source_guid

The application logical data source associated with the sample.

property source_timestamp

The DataWriter's write timestamp.

property state

Get the DataState of the sample.

property topic_query_guid

The GUID of the TopicQuery that is related to the sample.

property valid

Indicates whether the DataSample contains data or else it is only used to communicate a change in the InstanceState of the instance.

class `rti.connextdds.SampleLostState`

Bases: `pybind11_object`

`LOST_BY_AVAILABILITY_WAITING_TIME = <rti.connextdds.SampleLostState object>`

`LOST_BY_DECODE_FAILURE = <rti.connextdds.SampleLostState object>`

`LOST_BY_DESERIALIZATION_FAILURE = <rti.connextdds.SampleLostState object>`

`LOST_BY_INCOMPLETE_COHERENT_SET = <rti.connextdds.SampleLostState object>`

`LOST_BY_INSTANCES_LIMIT = <rti.connextdds.SampleLostState object>`

`LOST_BY_LARGE_COHERENT_SET = <rti.connextdds.SampleLostState object>`

`LOST_BY_OUT_OF_MEMORY = <rti.connextdds.SampleLostState object>`

`LOST_BY_REMOTE_WRITERS_PER_INSTANCE_LIMIT = <rti.connextdds.SampleLostState object>`

`LOST_BY_REMOTE_WRITERS_PER_SAMPLE_LIMIT = <rti.connextdds.SampleLostState object>`

`LOST_BY_REMOTE_WRITERS_PER_VIRTUAL_QUEUE_LIMIT = <rti.connextdds.SampleLostState object>`

`LOST_BY_SAMPLES_LIMIT = <rti.connextdds.SampleLostState object>`

`LOST_BY_SAMPLES_PER_INSTANCE_LIMIT = <rti.connextdds.SampleLostState object>`

`LOST_BY_SAMPLES_PER_REMOTE_WRITER_LIMIT = <rti.connextdds.SampleLostState object>`

`LOST_BY_UNKNOWN_INSTANCE = <rti.connextdds.SampleLostState object>`

`LOST_BY_VIRTUAL_WRITERS_LIMIT = <rti.connextdds.SampleLostState object>`

`LOST_BY_WRITER = <rti.connextdds.SampleLostState object>`

`NOT_LOST = <rti.connextdds.SampleLostState object>`

__and__ (*self*: rti.connexdds.SampleLostState, *arg0*: rti.connexdds.SampleLostState) → *rti.connexdds.SampleLostState*
 Bitwise logical AND of masks.

__bool__ (*self*: rti.connexdds.SampleLostState) → bool
 Test if any bits are set.

__contains__ (*self*: rti.connexdds.SampleLostState, *arg0*: rti.connexdds.SampleLostState) → bool

__eq__ (*self*: rti.connexdds.SampleLostState, *arg0*: rti.connexdds.SampleLostState) → bool
 Compare masks for equality.

__getitem__ (*self*: rti.connexdds.SampleLostState, *arg0*: int) → bool
 Get individual mask bit.

__hash__ = None

__iand__ (*self*: rti.connexdds.SampleLostState, *arg0*: rti.connexdds.SampleLostState) → *rti.connexdds.SampleLostState*
 Set mask to logical AND with another mask.

__ilshift__ (*self*: rti.connexdds.SampleLostState, *arg0*: int) → *rti.connexdds.SampleLostState*
 Left shift bits in mask.

__init__ (*self*: rti.connexdds.SampleLostState) → None
 Creates SampleLostState.NOT_LOST

__int__ (*self*: rti.connexdds.SampleLostState) → int
 Convert mask to int.

__ior__ (*self*: rti.connexdds.SampleLostState, *arg0*: rti.connexdds.SampleLostState) → *rti.connexdds.SampleLostState*
 Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.SampleLostState, *arg0*: int) → *rti.connexdds.SampleLostState*
 Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.SampleLostState, *arg0*: rti.connexdds.SampleLostState) → *rti.connexdds.SampleLostState*
 Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.SampleLostState, *arg0*: int) → *rti.connexdds.SampleLostState*
 Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.SampleLostState, *arg0*: rti.connexdds.SampleLostState) → bool
 Compare masks for inequality.

__or__ (*self*: rti.connexdds.SampleLostState, *arg0*: rti.connexdds.SampleLostState) → *rti.connexdds.SampleLostState*

Bitwise logical OR of masks.

__rshift__ (*self*: rti.connexdds.SampleLostState, *arg0*: int) → *rti.connexdds.SampleLostState*

Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.SampleLostState, *arg0*: int, *arg1*: bool) → None

Set individual mask bit

__str__ (*self*: rti.connexdds.SampleLostState) → str

__xor__ (*self*: rti.connexdds.SampleLostState, *arg0*: rti.connexdds.SampleLostState) → *rti.connexdds.SampleLostState*

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (**args*, ***kwargs*)

Overloaded function.

1. flip(*self*: rti.connexdds.SampleLostState) -> rti.connexdds.SampleLostState

Flip all bits in the mask.

2. flip(*self*: rti.connexdds.SampleLostState, *pos*: int) -> rti.connexdds.SampleLostState

Flip the mask bit at the specified position.

reset (**args*, ***kwargs*)

Overloaded function.

1. reset(*self*: rti.connexdds.SampleLostState) -> rti.connexdds.SampleLostState

Clear all bits in the mask.

2. reset(*self*: rti.connexdds.SampleLostState, *pos*: int) -> rti.connexdds.SampleLostState

Clear the mask bit at the specified position.

set (**args*, ***kwargs*)

Overloaded function.

1. set(*self*: rti.connexdds.SampleLostState) -> rti.connexdds.SampleLostState

Set all bits in the mask.

2. set(*self*: rti.connexdds.SampleLostState, *pos*: int, *value*: bool = True) -> rti.connexdds.SampleLostState

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (*self*: rti.connextdds.SampleLostState, *pos*: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (*self*: rti.connextdds.SampleLostState) → bool

Test if all bits are set.

test_any (*self*: rti.connextdds.SampleLostState) → bool

Test if any bits are set.

test_none (*self*: rti.connextdds.SampleLostState) → bool

Test if none of the bits are set.

class rti.connextdds.SampleLostStatus

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connextdds'

property last_reason

The reason for the most recent sample loss.

property total_count

Total count of all samples lost across all instances of data published under the Topic.

property total_count_change

The incremental number of samples lost since the last time the listener was called or the status was read.

class rti.connextdds.SampleRejectedState

Bases: pybind11_object

NOT_REJECTED = <rti.connextdds.SampleRejectedState object>

REJECTED_BY_DECODE_FAILURE = <rti.connextdds.SampleRejectedState object>

REJECTED_BY_INSTANCES_LIMIT = <rti.connextdds.SampleRejectedState object>

REJECTED_BY_REMOTE_WRITERS_PER_VIRTUAL_QUEUE_LIMIT = <rti.connextdds.SampleRejectedState object>

REJECTED_BY_SAMPLES_LIMIT = <rti.connextdds.SampleRejectedState object>

REJECTED_BY_SAMPLES_PER_INSTANCE_LIMIT = <rti.connextdds.SampleRejectedState object>

REJECTED_BY_SAMPLES_PER_REMOTE_WRITER_LIMIT = <rti.connextdds.SampleRejectedState object>

__and__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: rti.connexdds.SampleRejectedState) → *rti.connexdds.SampleRejectedState*

Bitwise logical AND of masks.

__bool__ (*self*: rti.connexdds.SampleRejectedState) → bool

Test if any bits are set.

__contains__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: rti.connexdds.SampleRejectedState) → bool

__eq__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: rti.connexdds.SampleRejectedState) → bool

Compare masks for equality.

__getitem__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: int) → bool

Get individual mask bit.

__hash__ = None

__iand__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: rti.connexdds.SampleRejectedState) → *rti.connexdds.SampleRejectedState*

Set mask to logical AND with another mask.

__ilshift__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: int) → *rti.connexdds.SampleRejectedState*

Left shift bits in mask.

__init__ (*self*: rti.connexdds.SampleRejectedState) → None

Creates SampleRejectedState.NOT_REJECTED

__int__ (*self*: rti.connexdds.SampleRejectedState) → int

Convert mask to int.

__ior__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: rti.connexdds.SampleRejectedState) → *rti.connexdds.SampleRejectedState*

Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: int) → *rti.connexdds.SampleRejectedState*

Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: rti.connexdds.SampleRejectedState) → *rti.connexdds.SampleRejectedState*

Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: int) → *rti.connexdds.SampleRejectedState*

Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: rti.connexdds.SampleRejectedState) → bool

Compare masks for inequality.

__or__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: rti.connexdds.SampleRejectedState) → *rti.connexdds.SampleRejectedState*

Bitwise logical OR of masks.

__rshift__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: int) → *rti.connexdds.SampleRejectedState*

Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: int, *arg1*: bool) → None

Set individual mask bit

__str__ (*self*: rti.connexdds.SampleRejectedState) → str

__xor__ (*self*: rti.connexdds.SampleRejectedState, *arg0*: rti.connexdds.SampleRejectedState) → *rti.connexdds.SampleRejectedState*

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (*args, **kwargs)

Overloaded function.

1. flip(*self*: rti.connexdds.SampleRejectedState) -> rti.connexdds.SampleRejectedState

Flip all bits in the mask.

2. flip(*self*: rti.connexdds.SampleRejectedState, *pos*: int) -> rti.connexdds.SampleRejectedState

Flip the mask bit at the specified position.

reset (*args, **kwargs)

Overloaded function.

1. reset(*self*: rti.connexdds.SampleRejectedState) -> rti.connexdds.SampleRejectedState

Clear all bits in the mask.

2. reset(*self*: rti.connexdds.SampleRejectedState, *pos*: int) -> rti.connexdds.SampleRejectedState

Clear the mask bit at the specified position.

set (*args, **kwargs)

Overloaded function.

1. set(*self*: rti.connexdds.SampleRejectedState) -> rti.connexdds.SampleRejectedState

Set all bits in the mask.

2. `set(self: rti.connextdds.SampleRejectedState, pos: int, value: bool = True) -> rti.connextdds.SampleRejectedState`

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (*self*: rti.connextdds.SampleRejectedState, *pos*: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (*self*: rti.connextdds.SampleRejectedState) → bool

Test if all bits are set.

test_any (*self*: rti.connextdds.SampleRejectedState) → bool

Test if any bits are set.

test_none (*self*: rti.connextdds.SampleRejectedState) → bool

Test if none of the bits are set.

class rti.connextdds.SampleRejectedStatus

Bases: pybind11_object

__init__ (*args, **kwargs)

__module__ = 'rti.connextdds'

property last_instance_handle

Handle for the instance of the sample that was most recently rejected.

property last_reason

Reason for the DataReader’s most recent sample rejection.

property total_count

Total count of samples rejected by the DataReader.

property total_count_change

The delta number of samples rejected since the last time the listener fired or the status was read.

class rti.connextdds.SampleState

Bases: pybind11_object

ANY = <rti.connextdds.SampleState object>

NOT_READ = <rti.connextdds.SampleState object>

READ = <rti.connextdds.SampleState object>

__and__ (*self*: rti.connextdds.SampleState, *arg0*: rti.connextdds.SampleState) → rti.connextdds.SampleState

Bitwise logical AND of masks.

__bool__ (*self*: rti.connexdds.SampleState) → *rti.connexdds.SampleState*
 Test if any bits are set.

__contains__ (*self*: rti.connexdds.SampleState, *arg0*: rti.connexdds.SampleState) → bool

__eq__ (*self*: rti.connexdds.SampleState, *arg0*: rti.connexdds.SampleState) → bool
 Compare masks for equality.

__getitem__ (*self*: rti.connexdds.SampleState, *arg0*: int) → bool
 Get individual mask bit.

__hash__ = None

__iand__ (*self*: rti.connexdds.SampleState, *arg0*: rti.connexdds.SampleState) → *rti.connexdds.SampleState*
 Set mask to logical AND with another mask.

__ilshift__ (*self*: rti.connexdds.SampleState, *arg0*: int) → *rti.connexdds.SampleState*
 Left shift bits in mask.

__init__ (**args*, ***kwargs*)
 Overloaded function.

1. **__init__**(*self*: rti.connexdds.SampleState) -> None
 Create a SampleState with no bits set.
2. **__init__**(*self*: rti.connexdds.SampleState, *value*: int) -> None
 Creates a mask from the bits in an integer.

__int__ (*self*: rti.connexdds.SampleState) → int
 Convert mask to int.

__ior__ (*self*: rti.connexdds.SampleState, *arg0*: rti.connexdds.SampleState) → *rti.connexdds.SampleState*
 Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.SampleState, *arg0*: int) → *rti.connexdds.SampleState*
 Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.SampleState, *arg0*: rti.connexdds.SampleState) → *rti.connexdds.SampleState*
 Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.SampleState, *arg0*: int) → *rti.connexdds.SampleState*
 Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.SampleState, *arg0*: rti.connexdds.SampleState) → bool
 Compare masks for inequality.

__or__ (*self*: rti.connexdds.SampleState, *arg0*: rti.connexdds.SampleState) → *rti.connexdds.SampleState*

Bitwise logical OR of masks.

__rshift__ (*self*: rti.connexdds.SampleState, *arg0*: int) → *rti.connexdds.SampleState*

Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.SampleState, *arg0*: int, *arg1*: bool) → None

Set individual mask bit

__str__ (*self*: rti.connexdds.SampleState) → str

__xor__ (*self*: rti.connexdds.SampleState, *arg0*: rti.connexdds.SampleState) → *rti.connexdds.SampleState*

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (*args, **kwargs)

Overloaded function.

1. flip(*self*: rti.connexdds.SampleState) -> rti.connexdds.SampleState

Flip all bits in the mask.

2. flip(*self*: rti.connexdds.SampleState, pos: int) -> rti.connexdds.SampleState

Flip the mask bit at the specified position.

reset (*args, **kwargs)

Overloaded function.

1. reset(*self*: rti.connexdds.SampleState) -> rti.connexdds.SampleState

Clear all bits in the mask.

2. reset(*self*: rti.connexdds.SampleState, pos: int) -> rti.connexdds.SampleState

Clear the mask bit at the specified position.

set (*args, **kwargs)

Overloaded function.

1. set(*self*: rti.connexdds.SampleState) -> rti.connexdds.SampleState

Set all bits in the mask.

2. set(*self*: rti.connexdds.SampleState, pos: int, value: bool = True) -> rti.connexdds.SampleState

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (*self*: rti.connexdds.SampleState, *pos*: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (*self*: rti.connexdds.SampleState) → bool

Test if all bits are set.

test_any (*self*: rti.connexdds.SampleState) → bool

Test if any bits are set.

test_none (*self*: rti.connexdds.SampleState) → bool

Test if none of the bits are set.

class rti.connexdds.SequenceNumber

Bases: pybind11_object

__add__ (*self*: rti.connexdds.SequenceNumber, *arg0*: rti.connexdds.SequenceNumber) → rti.connexdds.SequenceNumber

Add two SequenceNumbers.

__eq__ (*self*: rti.connexdds.SequenceNumber, *arg0*: rti.connexdds.SequenceNumber) → bool

Compare SequenceNumbers for equality.

__ge__ (*self*: rti.connexdds.SequenceNumber, *arg0*: rti.connexdds.SequenceNumber) → bool

Compare two SequenceNumbers for a greater-than-or-equal relationship.

__gt__ (*self*: rti.connexdds.SequenceNumber, *arg0*: rti.connexdds.SequenceNumber) → bool

Compare two SequenceNumbers for a greater-than relationship.

__hash__ = None

__iadd__ (*self*: rti.connexdds.SequenceNumber, *arg0*: rti.connexdds.SequenceNumber) → rti.connexdds.SequenceNumber

Add a SequenceNumber to another

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.SequenceNumber) -> None

Create a default SequenceNumber, equal to SequenceNumber.unknown

2. **__init__**(self: rti.connexdds.SequenceNumber, value: int) -> None

Creates a SequenceNumber from an integer.

__int__ (*self*: rti.connexdds.SequenceNumber) → int

Convert SequenceNumber to integer.

__isub__ (*self*: rti.connexdds.SequenceNumber, *arg0*: rti.connexdds.SequenceNumber) → rti.connexdds.SequenceNumber

Subtract a SequenceNumber from another.

`__le__` (*self*: rti.connexdds.SequenceNumber, *arg0*: rti.connexdds.SequenceNumber) → bool
 Compare two SequenceNumbers for a less-than-or-equal relationship.

`__lt__` (*self*: rti.connexdds.SequenceNumber, *arg0*: rti.connexdds.SequenceNumber) → bool
 Compare two SequenceNumbers for a less-than relationship.

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.SequenceNumber, *arg0*: rti.connexdds.SequenceNumber) → bool
 Compare SequenceNumbers for inequality.

`__repr__` (*self*: rti.connexdds.SequenceNumber) → str
 Convert SequenceNumber to string.

`__sub__` (*self*: rti.connexdds.SequenceNumber, *arg0*: rti.connexdds.SequenceNumber) →
rti.connexdds.SequenceNumber
 Subtract one SequenceNumber from another.

`automatic` = -1

`maximum` = 9223372036854775807

`unknown` = -1

property value

Get/set the SequenceNumber value.

`zero` = 0

class rti.connexdds.SequenceType

Bases: *UnidimensionalCollectionType*

`__eq__` (*self*: rti.connexdds.SequenceType, *arg0*: rti.connexdds.SequenceType) → bool
 Test for equality.

`__hash__` = None

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.SequenceType, data_type: rti.connexdds.DynamicType) ->`
 None

Creates an unbounded collection with an element type.

2. `__init__(self: rti.connexdds.SequenceType, data_type: rti.connexdds.DynamicType,`
`bounds: int) ->` None

Creates a bounded collection with an element type.

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.SequenceType, *arg0*: rti.connexdds.SequenceType) → bool
 Test for inequality.

```
class rti.connexdds.Service
```

```
Bases: pybind11_object
```

```
__eq__ (self: rti.connexdds.Service, arg0: rti.connexdds.Service) → bool
```

```
Test for equality.
```

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

```
Overloaded function.
```

```
1. __init__(self: rti.connexdds.Service) -> None
```

```
Creates the default policy (no service).
```

```
2. __init__(self: rti.connexdds.Service, kind: rti.connexdds.ServiceKind) -> None
```

```
Creates an instance with the specified service kind.
```

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.Service, arg0: rti.connexdds.Service) → bool
```

```
Test for inequality.
```

```
property kind
```

```
The service kind.
```

```
class rti.connexdds.ServiceKind
```

```
Bases: pybind11_object
```

```
DATABASE_INTEGRATION = <ServiceKind.DATABASE_INTEGRATION: 6>
```

```
NO_SERVICE = <ServiceKind.NO_SERVICE: 0>
```

```
PERSISTENCE = <ServiceKind.PERSISTENCE: 1>
```

```
QUEUING = <ServiceKind.QUEUING: 2>
```

```
RECORDING = <ServiceKind.RECORDING: 4>
```

```
REPLAY = <ServiceKind.REPLAY: 5>
```

```
ROUTING = <ServiceKind.ROUTING: 3>
```

```
class ServiceKind
```

```
Bases: pybind11_object
```

```
Members:
```

```
NO_SERVICE : There is no service associated to the Entity.
```

```
PERSISTENCE : The Entity is associated to RTI Persistence Service.
```

```
QUEUING : The Entity is associated to RTI Queuing Service.
```

```
ROUTING : The Entity is associated to RTI Routing Service.
```

RECORDING : The Entity is associated to RTI Recording Service.

REPLAY : The Entity is associated to RTI Replay Service.

DATABASE_INTEGRATION : The Entity is associated to RTI Database Integration Service.

WEB_INTEGRATION : The Entity is associated to RTI Web Integration Service.

DATABASE_INTEGRATION = <ServiceKind.DATABASE_INTEGRATION: 6>

NO_SERVICE = <ServiceKind.NO_SERVICE: 0>

PERSISTENCE = <ServiceKind.PERSISTENCE: 1>

QUEUING = <ServiceKind.QUEUING: 2>

RECORDING = <ServiceKind.RECORDING: 4>

REPLAY = <ServiceKind.REPLAY: 5>

ROUTING = <ServiceKind.ROUTING: 3>

WEB_INTEGRATION = <ServiceKind.WEB_INTEGRATION: 7>

`__eq__` (*self: object, other: object*) → bool

`__getstate__` (*self: object*) → int

`__hash__` (*self: object*) → int

`__index__` (*self: rti.connexdds.ServiceKind.ServiceKind*) → int

`__init__` (*self: rti.connexdds.ServiceKind.ServiceKind, value: int*) → None

`__int__` (*self: rti.connexdds.ServiceKind.ServiceKind*) → int

```
__members__ = {'DATABASE_INTEGRATION':
<ServiceKind.DATABASE_INTEGRATION: 6>, 'NO_SERVICE':
<ServiceKind.NO_SERVICE: 0>, 'PERSISTENCE':
<ServiceKind.PERSISTENCE: 1>, 'QUEUING': <ServiceKind.QUEUING:
2>, 'RECORDING': <ServiceKind.RECORDING: 4>, 'REPLAY':
<ServiceKind.REPLAY: 5>, 'ROUTING': <ServiceKind.ROUTING: 3>,
'WEB_INTEGRATION': <ServiceKind.WEB_INTEGRATION: 7>}
```

`__module__` = 'rti.connexdds'

`__ne__` (*self: object, other: object*) → bool

`__repr__` (*self: object*) → str

`__setstate__` (*self: rti.connexdds.ServiceKind.ServiceKind, state: int*) → None

`__str__` ()

name(self: handle) -> str

property name

property value

WEB_INTEGRATION = <ServiceKind.WEB_INTEGRATION: 7>

__eq__ (*self*: rti.connextdds.ServiceKind, *arg0*: rti.connextdds.ServiceKind) → bool
Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connextdds.ServiceKind, *arg0*: rti.connextdds.ServiceKind) → bool
Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connextdds.ServiceKind, *arg0*: rti.connextdds.ServiceKind) → bool
Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.ServiceKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connextdds.ServiceKind, *arg0*: rti.connextdds.ServiceKind.ServiceKind) -> None

Copy constructor.

__int__ (*self*: rti.connextdds.ServiceKind) → *rti.connextdds.ServiceKind.ServiceKind*
Int conversion.

__le__ (*self*: rti.connextdds.ServiceKind, *arg0*: rti.connextdds.ServiceKind) → bool
Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connextdds.ServiceKind, *arg0*: rti.connextdds.ServiceKind) → bool
Apply operator to underlying enumerated values.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.ServiceKind, *arg0*: rti.connextdds.ServiceKind) → bool
Apply operator to underlying enumerated values.

__str__ (*self*: rti.connextdds.ServiceKind) → str
String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connextdds.**ServiceRequest**

Bases: *pybind11_object*

class **ContentFilter**

Bases: *ContentFilterBase*

__init__ (*self*: rti.connexdds.ServiceRequest.ContentFilter) → None

__module__ = 'rti.connexdds'

compile (*self*: rti.connexdds.ServiceRequest.ContentFilter, *expression*: str, *parameters*: rti.connexdds.StringSeq, *type_code*: Optional[rti.connexdds.DynamicType], *type_class_name*: str, *old_compile_data*: Optional[object]) → Optional[object]

Compile an instance of the content filter according to the filter expression and parameters of the given data type.

evaluate (*self*: rti.connexdds.ServiceRequest.ContentFilter, *compile_data*: Optional[object], *sample*: rti.connexdds.ServiceRequest, *meta_data*: rti.connexdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

finalize (*self*: rti.connexdds.ServiceRequest.ContentFilter, *compile_data*: Optional[object]) → None

A previously compiled instance of the content filter is no longer in use and resources can now be cleaned up.

class ContentFilteredTopic

Bases: *ITopicDescription*, *IAnyTopic*

__eq__ (*self*: rti.connexdds.ServiceRequest.ContentFilteredTopic, *arg0*: rti.connexdds.ServiceRequest.ContentFilteredTopic) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ServiceRequest.ContentFilteredTopic, *topic*: rti.connexdds.ServiceRequest.Topic, *name*: str, *contentfilter*: rti.connexdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. **__init__**(*self*: rti.connexdds.ServiceRequest.ContentFilteredTopic, *topic_description*: rti.connexdds.ServiceRequest.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ServiceRequest.ContentFilteredTopic, *arg0*: rti.connexdds.ServiceRequest.ContentFilteredTopic) → bool

Test for inequality.

append_to_expression_parameter (*self*: rti.connexdds.ServiceRequest.ContentFilteredTopic, *index*: int, *extension*: str) → None

Append the extension to the end of parameter at the provided index, separated by a comma.

property filter_expression

Get the filter expression

property filter_parameters

Get/set the filter parameters.

static find (*participant*: rti.connextdds.DomainParticipant, *name*: str) → Optional[rti.connextdds.ServiceRequest.ContentFilteredTopic]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

remove_from_expression_parameter (*self*: rti.connextdds.ServiceRequest.ContentFilteredTopic, *index*: int, *remove_term*: str) → None

Removes the specified term from the parameter at the provided index.

set_filter (*self*: rti.connextdds.ServiceRequest.ContentFilteredTopic, *arg0*: rti.connextdds.Filter) → None

Set the filter.

property topic

Get the underlying Topic.

class ContentFilteredTopicSeq

Bases: pybind11_object

__add__ (*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *arg0*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) → rti.connextdds.ServiceRequest.ContentFilteredTopicSeq

__bool__ (*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *x*: rti.connextdds.ServiceRequest.ContentFilteredTopic) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *arg0*: int) -> None

Delete the list elements at index i

2. **__delitem__**(*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *arg0*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__(self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, s: slice) -> rti.connexdds.ServiceRequest.ContentFilteredTopicSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, arg0: int) -> rti.connexdds.ServiceRequest.ContentFilteredTopic`

`__hash__ = None`

`__iadd__ (self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, arg0: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq) -> rti.connexdds.ServiceRequest.ContentFilteredTopicSeq`

`__imul__ (self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, arg0: int) -> rti.connexdds.ServiceRequest.ContentFilteredTopicSeq`

`__init__ (*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq) -> None`

2. `__init__(self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, arg0: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, arg0: Iterable) -> None`

`__iter__ (self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq) -> Iterator`

`__len__ (self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq) -> int`

`__module__ = 'rti.connexdds'`

`__mul__ (self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, arg0: int) -> rti.connexdds.ServiceRequest.ContentFilteredTopicSeq`

`__ne__ (self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, arg0: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq) -> bool`

`__rmul__ (self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, arg0: int) -> rti.connexdds.ServiceRequest.ContentFilteredTopicSeq`

`__setitem__ (*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, arg0: int, arg1: rti.connexdds.ServiceRequest.ContentFilteredTopic) -> None`

2. `__setitem__(self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, arg0: slice, arg1: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq) -> None`

Assign list elements using a slice object

`append (self: rti.connexdds.ServiceRequest.ContentFilteredTopicSeq, x: rti.connexdds.ServiceRequest.ContentFilteredTopic) -> None`

Add an item to the end of the list

clear (*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) → None

Clear the contents

count (*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *x*:
rti.connextdds.ServiceRequest.ContentFilteredTopic) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *L*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) -> None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *i*: int, *x*:
rti.connextdds.ServiceRequest.ContentFilteredTopic) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq) -> rti.connextdds.ServiceRequest.ContentFilteredTopic

Remove and return the last item

2. pop(*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *i*: int) -> rti.connextdds.ServiceRequest.ContentFilteredTopic

Remove and return the item at index *i*

remove (*self*: rti.connextdds.ServiceRequest.ContentFilteredTopicSeq, *x*:
rti.connextdds.ServiceRequest.ContentFilteredTopic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class DataReader

Bases: *IDataReader*

class Selector

Bases: *pybind11_object*

__init__ (*self*: rti.connextdds.ServiceRequest.DataReader.Selector, *datareader*:
rti.connextdds.ServiceRequest.DataReader) → None

Create a Selector for a DataReader to read/take based on selected conditions

__module__ = 'rti.connextdds'

condition (*self*: rti.connextdds.ServiceRequest.DataReader.Selector, *condition*:
rti.connextdds.IReadCondition) →
rti.connextdds.ServiceRequest.DataReader.Selector

Select samples based on a ReadCondition.

content (*self*: rti.connextdds.ServiceRequest.DataReader.Selector, *query*: rti.connextdds.Query) → *rti.connextdds.ServiceRequest.DataReader.Selector*

Select samples based on a Query.

instance (*self*: rti.connextdds.ServiceRequest.DataReader.Selector, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.ServiceRequest.DataReader.Selector*

Select a specific instance to read/take.

max_samples (*self*: rti.connextdds.ServiceRequest.DataReader.Selector, *max*: int) → *rti.connextdds.ServiceRequest.DataReader.Selector*

Limit the number of samples read/taken by the Select.

next_instance (*self*: rti.connextdds.ServiceRequest.DataReader.Selector, *previous*: rti.connextdds.InstanceHandle) → *rti.connextdds.ServiceRequest.DataReader.Selector*

Select the instance after the specified instance to read/take.

read (*self*: rti.connextdds.ServiceRequest.DataReader.Selector) → list

Read copies of available samples (data and info) based on the Selector settings.

read_data (*self*: rti.connextdds.ServiceRequest.DataReader.Selector) → list

Read copies of available valid data based on the Selector settings.

read_loaned (*self*: rti.connextdds.ServiceRequest.DataReader.Selector) → *rti.connextdds.ServiceRequest.LoanedSamples*

Take available samples (data and info) based on the Selector settings and return them in a loaned container.

state (*self*: rti.connextdds.ServiceRequest.DataReader.Selector, *state*: rti.connextdds.DataState) → *rti.connextdds.ServiceRequest.DataReader.Selector*

Select samples with a specified data state.

take (*self*: rti.connextdds.ServiceRequest.DataReader.Selector) → list

Take copies of available samples (data and info) based on the Selector settings.

take_data (*self*: rti.connextdds.ServiceRequest.DataReader.Selector) → list

Take copies of available valid data based on the Selector settings.

take_loaned (*self*: rti.connextdds.ServiceRequest.DataReader.Selector) → *rti.connextdds.ServiceRequest.LoanedSamples*

Read available samples (data and info) based on the Selector settings and return them in a loaned container.

__enter__ (*self*: rti.connextdds.ServiceRequest.DataReader) → *rti.connextdds.ServiceRequest.DataReader*

Enter a context for this DataReader, to be cleaned up on exiting context

__eq__ (*self*: rti.connextdds.ServiceRequest.DataReader, *arg0*: rti.connextdds.ServiceRequest.DataReader) → bool

Test for equality.

__exit__ (*self*: rti.connexdds.ServiceRequest.DataReader, *arg0*: object, *arg1*: object, *arg2*: object) → None

Exit the context for this DataReader, cleaning up resources.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReader, *topic*: rti.connexdds.ServiceRequest.Topic) → None

Create a DataReader in the implicit subscriber with default QoS.

2. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReader, *topic*: rti.connexdds.ServiceRequest.Topic, *qos*: rti.connexdds.DataReaderQos, *listener*: rti.connexdds.ServiceRequest.DataReaderListener = None, *mask*: rti.connexdds.StatusMask = StatusMask.ALL) → None

Create a DataReader in the implicit subscriber with specific QoS and a listener.

3. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReader, *cft*: rti.connexdds.ServiceRequest.ContentFilteredTopic) → None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with default QoS.

4. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReader, *cft*: rti.connexdds.ServiceRequest.ContentFilteredTopic, *qos*: rti.connexdds.DataReaderQos, *listener*: rti.connexdds.ServiceRequest.DataReaderListener = None, *mask*: rti.connexdds.StatusMask = StatusMask.ALL) → None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with specific QoS.

5. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReader, *subscriber*: rti.connexdds.Subscriber, *topic*: rti.connexdds.ServiceRequest.Topic) → None

Create a DataReader.

6. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReader, *subscriber*: rti.connexdds.Subscriber, *topic*: rti.connexdds.ServiceRequest.Topic, *qos*: rti.connexdds.DataReaderQos, *listener*: rti.connexdds.ServiceRequest.DataReaderListener = None, *mask*: rti.connexdds.StatusMask = StatusMask.ALL) → None

Create a DataReader in a subscriber with specific QoS and a listener.

7. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReader, *subscriber*: rti.connexdds.Subscriber, *cft*: rti.connexdds.ServiceRequest.ContentFilteredTopic) → None

Create a DataReader with a ContentFilteredTopic in a subscriber with default QoS.

8. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReader, *subscriber*: rti.connexdds.Subscriber, *cft*: rti.connexdds.ServiceRequest.ContentFilteredTopic, *qos*: rti.connexdds.DataReaderQos, *listener*: rti.connexdds.ServiceRequest.DataReaderListener = None, *mask*: rti.connexdds.StatusMask = StatusMask.ALL) → None

Create a DataReader with a ContentFilteredTopic in a subscriber with specific QoS.

9. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReader, *reader*: rti.connexdds.IAnyDataReader) → None

Get a typed DataReader from an AnyDataReader.

10. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReader, *entity*: rti.connexdds.IEntity) → None

Get a typed DataReader from an Entity.

__lshift__ (*self*: rti.connexdds.ServiceRequest.DataReader, *arg0*: rti.connexdds.DataReaderQos) → *rti.connexdds.ServiceRequest.DataReader*

Set the DataReaderQos for this DataReader.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ServiceRequest.DataReader, *arg0*: rti.connexdds.ServiceRequest.DataReader) → bool

Test for inequality.

__rshift__ (*self*: rti.connexdds.ServiceRequest.DataReader, *arg0*: rti.connexdds.DataReaderQos) → *rti.connexdds.ServiceRequest.DataReader*

Get the DataReaderQos from this DataReader

acknowledge_all (**args*, ***kwargs*)

Overloaded function.

1. `acknowledge_all(self: rti.connexdds.ServiceRequest.DataReader) -> None`

Acknowledge all previously accessed samples.

2. `acknowledge_all(self: rti.connexdds.ServiceRequest.DataReader, arg0: rti.connexdds.AckResponseData) -> None`

Acknowledge all previously accessed samples.

acknowledge_sample (**args*, ***kwargs*)

Overloaded function.

1. `acknowledge_sample(self: rti.connexdds.ServiceRequest.DataReader, sample_info: rti.connexdds.SampleInfo) -> None`

Acknowledge a single sample.

2. `acknowledge_sample(self: rti.connexdds.ServiceRequest.DataReader, sample_info: rti.connexdds.SampleInfo, ack_response_data: rti.connexdds.AckResponseData) -> None`

Acknowledge a single sample with ack response data.

close (*self*: rti.connexdds.ServiceRequest.DataReader) → None

Close this DataReader.

property datareader_cache_status

Get the DataReaderCacheStatus for the DataReader.

property datareader_protocol_status

Get the DataReaderProtocolStatus for the DataReader.

property default_filter_state

Returns the filter state for the read/take operations.

static find_all_by_topic (*subscriber*: rti.connexdds.Subscriber, *topic_name*: str) → *rti.connexdds.ServiceRequest.DataReaderSeq*

Retrieve all DataReaders for the given topic name in the subscriber.

static find_by_name (**args*, ***kwargs*)

Overloaded function.

1. `find_by_name(participant: rti.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.ServiceRequest.DataReader]`

Find DataReader in DomainParticipant with the DataReader's name, returning the first found.

2. `find_by_name(subscriber: rti.connextdds.Subscriber, name: str) -> Optional[rti.connextdds.ServiceRequest.DataReader]`

Find DataReader in Subscriber with the DataReader's name, returning the first found.

static find_by_topic (*subscriber: rti.connextdds.Subscriber, name: str*) → *Optional[rti.connextdds.ServiceRequest.DataReader]*

Find DataReader in Subscriber with a topic name, returning the first found.

is_matched_publication_alive (*self: rti.connextdds.ServiceRequest.DataReader, arg0: rti.connextdds.InstanceHandle*) → *bool*

Check if a matched publication is alive.

key_value (*self: rti.connextdds.ServiceRequest.DataReader, handle: rti.connextdds.InstanceHandle*) → *rti.connextdds.ServiceRequest*

Retrieve the instance key that corresponds to an instance handle.

property listener

Gets or sets the listener with StatusMask.ALL

property liveliness_changed_status

Get the LivelinessChangedStatus for this DataReader.

lookup_instance (*self: rti.connextdds.ServiceRequest.DataReader, key_holder: rti.connextdds.ServiceRequest*) → *rti.connextdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

matched_publication_data (*self: rti.connextdds.ServiceRequest.DataReader, handle: rti.connextdds.InstanceHandle*) → *rti.connextdds.PublicationBuiltinTopicData*

Get the PublicationBuiltinTopicData for a publication matched to this DataReader.

matched_publication_datareader_protocol_status (*self: rti.connextdds.ServiceRequest.DataReader, publication_handle: rti.connextdds.InstanceHandle*) → *rti.connextdds.DataReaderProtocolStatus*

Get the DataReaderProtocolStatus for the DataReader based on the matched publication handle.

matched_publication_participant_data (*self: rti.connextdds.ServiceRequest.DataReader, handle: rti.connextdds.InstanceHandle*) → *rti.connextdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a publication matched to this DataReader.

property matched_publications

Get a copy of the list of the currently matched publication handles.

property qos

The DataReaderQos for this DataReader.

This property's getter returns a deep copy.

read (*self*: rti.connextdds.ServiceRequest.DataReader) → list

Read copies of all available samples (data and info).

read_data (*self*: rti.connextdds.ServiceRequest.DataReader) → list

Read copies of all available valid data.

read_loaned (*self*: rti.connextdds.ServiceRequest.DataReader) → *rti.connextdds.ServiceRequest.LoanedSamples*

Read all available samples (data and info) and return them in a loaned container.

property requested_deadline_missed_status

Get the RequestedDeadlineMissed status for the DataReader

property requested_incompatible_qos_status

Get the RequestedIncompatibleQosStatus for the DataReader.

property sample_lost_status

Get the SampleLostStatus for the DataReader.

property sample_rejected_status

Get the SampleRejectedStatus for the DataReader.

select (*self*: rti.connextdds.ServiceRequest.DataReader) →

dds::sub::DataReader<rti::topic::ServiceRequest, rti::sub::DataReaderImpl>::Selector

Get a Selector to perform complex data selections, such as per-instance selection, content, and status filtering.

set_listener (*self*: rti.connextdds.ServiceRequest.DataReader, *listener*: *rti.connextdds.ServiceRequest.DataReaderListener*, *event_mask*: *rti.connextdds.StatusMask*) → None

Set the listener and associated event mask.

property subscriber

Returns the parent Subscriber of the DataReader.

property subscription_matched_status

Get the SubscriptionMatchedStatus for the DataReader.

take (*self*: rti.connextdds.ServiceRequest.DataReader) → list

Take copies of all available samples (data and info).

take_data (*self*: rti.connextdds.ServiceRequest.DataReader) → list

Take copies of all available valid data.

take_loaned (*self*: rti.connextdds.ServiceRequest.DataReader) →
rti.connextdds.ServiceRequest.LoanedSamples

Take all available samples (data and info) and return them in a loaned container.

property topic_description

Returns the TopicDescription associated with the DataReader.

property topic_name

Get the topic name associated with this DataReader.

property type_name

Get the type name associated with this DataReader.

wait_for_historical_data (*self*: rti.connextdds.ServiceRequest.DataReader,
max_wait: rti.connextdds.Duration) → None

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind.

wait_for_historical_data_async (*self*: rti.connextdds.ServiceRequest.DataReader,
max_wait: rti.connextdds.Duration) → object

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind. This call is awaitable and only for use with asyncio.

class DataReaderListener

Bases: pybind11_object

__init__ (*self*: rti.connextdds.ServiceRequest.DataReaderListener) → None

__module__ = 'rti.connextdds'

on_data_available (*self*: rti.connextdds.ServiceRequest.DataReaderListener, *arg0*:
rti.connextdds.ServiceRequest.DataReader) → None

Data available callback.

on_liveliness_changed (*self*: rti.connextdds.ServiceRequest.DataReaderListener, *arg0*:
rti.connextdds.ServiceRequest.DataReader, *arg1*:
rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*:
rti.connextdds.ServiceRequest.DataReaderListener,
arg0: rti.connextdds.ServiceRequest.DataReader,
arg1:
rti.connextdds.RequestedDeadlineMissedStatus) →
None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connextdds.ServiceRequest.DataReaderListener, *arg0*:
rti.connextdds.ServiceRequest.DataReader, *arg1*:
rti.connextdds.RequestedIncompatibleQosStatus)
→ None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connextdds.ServiceRequest.DataReaderListener, *arg0*: rti.connextdds.ServiceRequest.DataReader, *arg1*: rti.connextdds.SampleLostStatus) → None

Sample lost callback.

on_sample_rejected (*self*: rti.connextdds.ServiceRequest.DataReaderListener, *arg0*: rti.connextdds.ServiceRequest.DataReader, *arg1*: rti.connextdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connextdds.ServiceRequest.DataReaderListener, *arg0*: rti.connextdds.ServiceRequest.DataReader, *arg1*: rti.connextdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class DataReaderSeq

Bases: pybind11_object

__add__ (*self*: rti.connextdds.ServiceRequest.DataReaderSeq, *arg0*: rti.connextdds.ServiceRequest.DataReaderSeq) → rti.connextdds.ServiceRequest.DataReaderSeq

__bool__ (*self*: rti.connextdds.ServiceRequest.DataReaderSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connextdds.ServiceRequest.DataReaderSeq, *x*: rti.connextdds.ServiceRequest.DataReader) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connextdds.ServiceRequest.DataReaderSeq, *arg0*: int) → None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connextdds.ServiceRequest.DataReaderSeq, *arg0*: slice) → None

Delete list elements using a slice object

__eq__ (*self*: rti.connextdds.ServiceRequest.DataReaderSeq, *arg0*: rti.connextdds.ServiceRequest.DataReaderSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connextdds.ServiceRequest.DataReaderSeq, *s*: slice) → rti.connextdds.ServiceRequest.DataReaderSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connextdds.ServiceRequest.DataReaderSeq, *arg0*: int) → rti.connextdds.ServiceRequest.DataReader

__hash__ = None

__iadd__ (*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *arg0*: rti.connexdds.ServiceRequest.DataReaderSeq) → rti.connexdds.ServiceRequest.DataReaderSeq

__imul__ (*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *arg0*: int) → rti.connexdds.ServiceRequest.DataReaderSeq

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReaderSeq) -> None
2. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *arg0*: rti.connexdds.ServiceRequest.DataReaderSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.ServiceRequest.DataReaderSeq) → Iterator

__len__ (*self*: rti.connexdds.ServiceRequest.DataReaderSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *arg0*: int) → rti.connexdds.ServiceRequest.DataReaderSeq

__ne__ (*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *arg0*: rti.connexdds.ServiceRequest.DataReaderSeq) → bool

__rmul__ (*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *arg0*: int) → rti.connexdds.ServiceRequest.DataReaderSeq

__setitem__ (*args, **kwargs)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *arg0*: int, *arg1*: rti.connexdds.ServiceRequest.DataReader) -> None
2. **__setitem__**(*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *arg0*: slice, *arg1*: rti.connexdds.ServiceRequest.DataReaderSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *x*: rti.connexdds.ServiceRequest.DataReader) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.ServiceRequest.DataReaderSeq) → None

Clear the contents

count (*self*: rti.connexdds.ServiceRequest.DataReaderSeq, *x*: rti.connexdds.ServiceRequest.DataReader) → int

Return the number of times *x* appears in the list

extend (*args, **kwargs)

Overloaded function.

1. extend(self: rti.connextdds.ServiceRequest.DataReaderSeq, L: rti.connextdds.ServiceRequest.DataReaderSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.ServiceRequest.DataReaderSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.ServiceRequest.DataReaderSeq, i: int, x: rti.connextdds.ServiceRequest.DataReader) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connextdds.ServiceRequest.DataReaderSeq) -> rti.connextdds.ServiceRequest.DataReader

Remove and return the last item

2. pop(self: rti.connextdds.ServiceRequest.DataReaderSeq, i: int) -> rti.connextdds.ServiceRequest.DataReader

Remove and return the item at index i

remove (self: rti.connextdds.ServiceRequest.DataReaderSeq, x: rti.connextdds.ServiceRequest.DataReader) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class DataWriter

Bases: *IEntity, IAnyDataWriter*

__enter__ (self: rti.connextdds.ServiceRequest.DataWriter) → *rti.connextdds.ServiceRequest.DataWriter*

Enter a context for this DataWriter, to be cleaned up on exiting context

__eq__ (self: rti.connextdds.ServiceRequest.DataWriter, arg0: rti.connextdds.ServiceRequest.DataWriter) → bool

Test for equality.

__exit__ (self: rti.connextdds.ServiceRequest.DataWriter, arg0: object, arg1: object, arg2: object) → None

Exit the context for this DataWriter, cleaning up resources.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. __init__(self: rti.connextdds.ServiceRequest.DataWriter, topic: rti.connextdds.ServiceRequest.Topic) -> None

Creates a DataWriter in the implicit publisher with default QoS.

2. __init__(self: rti.connextdds.ServiceRequest.DataWriter, topic: rti.connextdds.ServiceRequest.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.Ser-

viceRequest.DataWriterListener = None, mask: rti.connexdds.StatusMask = StatusMask.ALL) -> None

Creates a DataWriter in the implicit publisher with specific QoS and optionally a listener.

3. `__init__(self: rti.connexdds.ServiceRequest.DataWriter, pub: rti.connexdds.Publisher, topic: rti.connexdds.ServiceRequest.Topic) -> None`

Creates a DataWriter in a publisher with default QoS.

4. `__init__(self: rti.connexdds.ServiceRequest.DataWriter, pub: rti.connexdds.Publisher, topic: rti.connexdds.ServiceRequest.Topic, qos: rti.connexdds.DataWriterQos, listener: rti.connexdds.ServiceRequest.DataWriterListener = None, mask: rti.connexdds.StatusMask = StatusMask.ALL) -> None`

Creates a DataWriter in a publisher with specific QoS and optionally a listener.

5. `__init__(self: rti.connexdds.ServiceRequest.DataWriter, writer: rti.connexdds.IAnyDataWriter) -> None`

Create a typed DataWriter from an AnyDataWriter.

6. `__init__(self: rti.connexdds.ServiceRequest.DataWriter, entity: rti.connexdds.IEntity) -> None`

Create a typed DataWriter from an Entity.

`__lshift__ (*args, **kwargs)`

Overloaded function.

1. `__lshift__(self: rti.connexdds.ServiceRequest.DataWriter, arg0: rti.connexdds.DataWriterQos) -> rti.connexdds.ServiceRequest.DataWriter`

Sets the DataWriterQos.

2. `__lshift__(self: rti.connexdds.ServiceRequest.DataWriter, arg0: Tuple[rti.connexdds.ServiceRequest, rti.connexdds.Time]) -> rti.connexdds.ServiceRequest.DataWriter`

Writes a paired sample with a timestamp.

3. `__lshift__(self: rti.connexdds.ServiceRequest.DataWriter, arg0: Tuple[rti.connexdds.ServiceRequest, rti.connexdds.InstanceHandle]) -> rti.connexdds.ServiceRequest.DataWriter`

Writes a paired sample with an instance handle.

4. `__lshift__(self: rti.connexdds.ServiceRequest.DataWriter, arg0: rti.connexdds.ServiceRequest.TimestampedSeq) -> rti.connexdds.ServiceRequest.DataWriter`

Writes a sequence of pairs of samples with timestamps.

5. `__lshift__(self: rti.connexdds.ServiceRequest.DataWriter, arg0: rti.connexdds.ServiceRequestSeq) -> rti.connexdds.ServiceRequest.DataWriter`

Writes a sequence of samples.

6. `__lshift__(self: rti.connexdds.ServiceRequest.DataWriter, arg0: rti.connexdds.ServiceRequest) -> rti.connexdds.ServiceRequest.DataWriter`

Writes a sample.

`__module__ = 'rti.connexdds'`

`__ne__(self: rti.connexdds.ServiceRequest.DataWriter, arg0: rti.connexdds.ServiceRequest.DataWriter) -> bool`

Test for inequality.

`__rshift__(self: rti.connexdds.ServiceRequest.DataWriter, arg0: rti.connexdds.DataWriterQos) -> rti.connexdds.ServiceRequest.DataWriter`

Get the DataWriterQos.

assert_liveliness (*self*: rti.connextdds.ServiceRequest.DataWriter) → None

Manually asserts the liveliness of the DataWriter.

close (*self*: rti.connextdds.ServiceRequest.DataWriter) → None

Close this DataWriter.

create_data (*self*: rti.connextdds.ServiceRequest.DataWriter) →
rti.connextdds.ServiceRequest

Create data of the writer's associated type and initialize it.

property datawriter_cache_status

Get a copy of the cache status for this writer.

property datawriter_protocol_status

Get a copy of the protocol status for this writer.

dispose_instance (**args, **kwargs*)

Overloaded function.

1. `dispose_instance(self: rti.connextdds.ServiceRequest.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.ServiceRequest.DataWriter`

Dispose an instance.

2. `dispose_instance(self: rti.connextdds.ServiceRequest.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.ServiceRequest.DataWriter`

Dispose an instance with a timestamp.

3. `dispose_instance(self: rti.connextdds.ServiceRequest.DataWriter, params: rti.connextdds.WriteParams) -> None`

Dispose an instance with params.

dispose_instance_async (**args, **kwargs*)

Overloaded function.

1. `dispose_instance_async(self: rti.connextdds.ServiceRequest.DataWriter, handle: rti.connextdds.InstanceHandle) -> object`

Dispose an instance.

2. `dispose_instance_async(self: rti.connextdds.ServiceRequest.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Dispose an instance with a timestamp.

3. `dispose_instance_async(self: rti.connextdds.ServiceRequest.DataWriter, key_holder: rti.connextdds.ServiceRequest) -> object`

Dispose the instance associated with `key_holder`.

4. `dispose_instance_async(self: rti.connextdds.ServiceRequest.DataWriter, key_holder: rti.connextdds.ServiceRequest, timestamp: rti.connextdds.Time) -> object`

Dispose the instance associated with `key_holder` using a timestamp

5. `dispose_instance_async(self: rti.connextdds.ServiceRequest.DataWriter, params: rti.connextdds.WriteParams) -> object`

Dispose an instance with params.

static find_all_by_topic (*publisher*: rti.connexdds.Publisher, *topic_name*: str) → *rti.connexdds.ServiceRequest.DataWriterSeq*

Retrieve all DataWriters for the given topic name in the publisher.

static find_by_name (**args*, ***kwargs*)

Overloaded function.

1. **find_by_name**(*participant*: rti.connexdds.DomainParticipant, *name*: str) → Optional[*rti.connexdds.ServiceRequest.DataWriter*]

Find DataWriter in DomainParticipant with the provided name, returning the first found.

2. **find_by_name**(*publisher*: rti.connexdds.Publisher, *name*: str) → Optional[*rti.connexdds.ServiceRequest.DataWriter*]

Find DataWriter in Publisher with the DataReader's name, returning the first found.

static find_by_topic (*publisher*: rti.connexdds.Publisher, *name*: str) → Optional[*rti.connexdds.ServiceRequest.DataWriter*]

Find DataWriter in publisher with a topic name, returning the first found.

flush (*self*: rti.connexdds.ServiceRequest.DataWriter) → None

Flushes the batch in progress in the context of thecalling thread.

is_matched_subscription_active (*self*: rti.connexdds.ServiceRequest.DataWriter, *arg0*: rti.connexdds.InstanceHandle) → bool

A boolean indicating whether or not the matched subscription is active.

is_sample_app_acknowledged (*self*: rti.connexdds.ServiceRequest.DataWriter, *sample_id*: rti.connexdds.SampleIdentity) → bool

Indicates if a sample is considered application-acknowledged.

key_value (*self*: rti.connexdds.ServiceRequest.DataWriter, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.ServiceRequest*

Retrieve the instance key that corresponds to an instance handle.

property listener

Get the listener associated with the DataWriter or set the listener.

property liveliness_lost_status

Get a copy of the LivelinessLostStatus.

lookup_instance (*self*: rti.connexdds.ServiceRequest.DataWriter, *key_holder*: rti.connexdds.ServiceRequest) → *rti.connexdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

matched_subscription_data (*self*: rti.connexdds.ServiceRequest.DataWriter, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.SubscriptionBuiltinTopicData*

Get the SubscriptionBuiltinTopicData for a subscription matched to this DataWriter.

matched_subscription_datawriter_protocol_status (**args*, ***kwargs*)

Overloaded function.

1. `matched_subscription_datawriter_protocol_status(self: rti.connextdds.ServiceRequest.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.DataWriterProtocolStatus`

Get a copy of the protocol status for this writer per a matched subscription handle.

2. `matched_subscription_datawriter_protocol_status(self: rti.connextdds.ServiceRequest.DataWriter, locator: rti.connextdds.Locator) -> rti.connextdds.DataWriterProtocolStatus`

Get a copy of the protocol status for this writer per a matched subscription locator.

matched_subscription_participant_data (*self: rti.connextdds.ServiceRequest.DataWriter, handle: rti.connextdds.InstanceHandle*) → *rti.connextdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a subscription matched to this DataWriter.

property matched_subscriptions

Get a copy of the list of the currently matched subscription handles.

property matched_subscriptions_locators

The locators used to communicate with matched DataReaders.

property offered_deadline_missed_status

Get a copy of the OfferedDeadlineMissedStatus.

property offered_incompatible_qos_status

Get a copy of the OfferedIncompatibleQosStatus

property publication_matched_status

Get a copy of the PublicationMatchedStatus

property publisher

Get the Publisher that owns this DataWriter.

property qos

The DataWriterQos for this DataWriter. This property's getter returns a deep copy.

register_instance (**args, **kwargs*)

Overloaded function.

1. `register_instance(self: rti.connextdds.ServiceRequest.DataWriter, key_holder: rti.connextdds.ServiceRequest) -> rti.connextdds.InstanceHandle`

Informs RTI Connex that the application will be modifying a particular instance.

2. `register_instance(self: rti.connextdds.ServiceRequest.DataWriter, key_holder: rti.connextdds.ServiceRequest, timestamp: rti.connextdds.Time) -> rti.connextdds.InstanceHandle`

Informs RTI Connex that the application will be modifying a particular instance and specified the timestamp.

3. `register_instance(self: rti.connextdds.ServiceRequest.DataWriter, key_holder: rti.connextdds.ServiceRequest, params: rti.connextdds.WriteParams) -> rti.connextdds.InstanceHandle`

Registers instance with parameters.

property reliable_reader_activity_changed_status

Get a copy of the reliable reader activity changed status for this writer.

property reliable_writer_cache_changed_status

Get a copy of the reliable cache status for this writer.

property service_request_accepted_status

Get a copy of the service request accepted status for this writer.

set_listener (*self*: rti.connextdds.ServiceRequest.DataWriter, *listener*: rti.connextdds.ServiceRequest.DataWriterListener, *event_mask*: rti.connextdds.StatusMask) → None

Set the listener and event mask for the DataWriter.

property topic

Get the Topic object associated with this DataWriter.

property topic_name

Get the topic name associated with this DataWriter.

property type_name

Get the type name for the topic object associated with this DataWriter.

unregister_instance (**args*, ***kwargs*)

Overloaded function.

1. `unregister_instance(self: rti.connextdds.ServiceRequest.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.ServiceRequest.DataWriter`

Unregister an instance.

2. `unregister_instance(self: rti.connextdds.ServiceRequest.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.ServiceRequest.DataWriter`

Unregister an instance with timestamp.

3. `unregister_instance(self: rti.connextdds.ServiceRequest.DataWriter, params: rti.connextdds.WriteParams) -> None`

Unregister an instance with parameters.

unregister_instance_async (**args*, ***kwargs*)

Overloaded function.

1. `unregister_instance_async(self: rti.connextdds.ServiceRequest.DataWriter, handle: rti.connextdds.InstanceHandle) -> object`

Unregister an instance.

2. `unregister_instance_async(self: rti.connextdds.ServiceRequest.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Unregister an instance with timestamp.

3. `unregister_instance_async(self: rti.connextdds.ServiceRequest.DataWriter, key_holder: rti.connextdds.ServiceRequest) -> object`

Unregister the instance associated with `key_holder`.

4. `unregister_instance_async(self: rti.connextdds.ServiceRequest.DataWriter, key_holder: rti.connextdds.ServiceRequest, timestamp: rti.connextdds.Time) -> object`

Unregister the instance associate with `key_holder` using a timestamp.

5. `unregister_instance_async(self: rti.connextdds.ServiceRequest.DataWriter, params: rti.connextdds.WriteParams) -> object`

Unregister an instance with parameters.

wait_for_acknowledgments (*self*: rti.connextdds.ServiceRequest.DataWriter, *max_wait*: rti.connextdds.Duration) → None

Blocks the calling thread until all data written by a reliable DataWriter is acknowledged or until the timeout expires.

wait_for_asynchronous_publishing (*self*: rti.connextdds.ServiceRequest.DataWriter, *max_wait*: rti.connextdds.Duration) → None

This operation blocks the calling thread (up to `max_wait`) until all data written by the asynchronous DataWriter is sent and acknowledged (if reliable) by all matched DataReader entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that `max_wait` elapsed before all the data was sent and/or acknowledged.

In other words, this guarantees that sending to best effort DataReader is complete in addition to what `DataWriter.wait_for_acknowledgments()` provides.

If the DataWriter does not have `PublishMode` kind set to `PublishModeKind.ASYNCHRONOUS` the operation will complete immediately

wait_for_asynchronous_publishing_async (*self*: rti.connextdds.ServiceRequest.DataWriter, *max_wait*: rti.connextdds.Duration) → object

This function is awaitable until either a timeout of `max_wait` or all data written by the asynchronous DataWriter is sent and acknowledged (if reliable) by all matched DataReader entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that `max_wait` elapsed before all the data was sent and/or acknowledged. This function works with `asyncio`.

In other words, this guarantees that sending to best effort DataReader is complete in addition to what `DataWriter.wait_for_acknowledgments()` provides.

If the DataWriter does not have `PublishMode` kind set to `PublishModeKind.ASYNCHRONOUS` the operation will complete immediately

write (**args*, ***kwargs*)

Overloaded function.

1. `write(self: rti.connextdds.ServiceRequest.DataWriter, samples: rti.connextdds.ServiceRequestSeq) -> None`

Write a sequence of samples.

2. `write(self: rti.connextdds.ServiceRequest.DataWriter, samples: rti.connextdds.ServiceRequestSeq, timestamp: rti.connextdds.Time) -> None`

Write a sequence of samples with a timestamp.

3. `write(self: rti.connextdds.ServiceRequest.DataWriter, sample: rti.connextdds.ServiceRequest) -> None`

Write a sample.

4. `write(self: rti.connextdds.ServiceRequest.DataWriter, sample: rti.connextdds.ServiceRequest, timestamp: rti.connextdds.Time) -> None`

Write a sample with a specified timestamp.

5. `write(self: rti.connextdds.ServiceRequest.DataWriter, sample: rti.connextdds.ServiceRequest, handle: rti.connextdds.InstanceHandle) -> None`

Write a sample with an instance handle.

6. `write(self: rti.connextdds.ServiceRequest.DataWriter, sample: rti.connextdds.ServiceRequest, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> None`

Write a sample with an instance handle and specified timestamp.

7. `write(self: rti.connextdds.ServiceRequest.DataWriter, sample: rti.connextdds.ServiceRequest, params: rti.connextdds.WriteParams) -> None`

Write with advanced parameters.

`write_async (*args, **kwargs)`

Overloaded function.

1. `write_async(self: rti.connextdds.ServiceRequest.DataWriter, sample: rti.connextdds.ServiceRequest) -> object`

Write a sample. This method is awaitable and is only for use with asyncio.

2. `write_async(self: rti.connextdds.ServiceRequest.DataWriter, sample: rti.connextdds.ServiceRequest, timestamp: rti.connextdds.Time) -> object`

Write a sample with a specified timestamp. This methods is awaitable and only for use with asyncio.

3. `write_async(self: rti.connextdds.ServiceRequest.DataWriter, sample: rti.connextdds.ServiceRequest, handle: rti.connextdds.InstanceHandle) -> object`

Write a sample with an instance handle. This method is awaitable and only for use with asyncio.

4. `write_async(self: rti.connextdds.ServiceRequest.DataWriter, sample: rti.connextdds.ServiceRequest, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Write a sample with an instance handle and specified timestamp. This method is awaitable and only for use with asyncio.

5. `write_async(self: rti.connextdds.ServiceRequest.DataWriter, samples: rti.connextdds.ServiceRequestSeq) -> object`

Write a sequence of samples. This method is awaitable and only for use with asyncio.

6. `write_async(self: rti.connextdds.ServiceRequest.DataWriter, samples: rti.connextdds.ServiceRequestSeq, timestamp: rti.connextdds.Time) -> object`

Write a sequence of samples with a timestamp. This method is awaitable and only for use with asyncio.

7. `write_async(self: rti.connextdds.ServiceRequest.DataWriter, samples: rti.connextdds.ServiceRequestSeq, handles: rti.connextdds.InstanceHandleSeq) -> object`

Write a sequence of samples with their instance handles. This method is awaitable and only for use with asyncio.

8. `write_async(self: rti.connextdds.ServiceRequest.DataWriter, samples: rti.connextdds.ServiceRequestSeq, handles: rti.connextdds.InstanceHandleSeq, timestamp: rti.connextdds.Time) -> object`

Write a sequence of samples with their instance handles and a timestamp. This method is awaitable and only for use with asyncio.

9. `write_async(self: rti.connextdds.ServiceRequest.DataWriter, sample: rti.connextdds.ServiceRequest, params: rti.connextdds.WriteParams) -> object`

Write with advanced parameters.

class DataWriterListener

Bases: `pybind11_object`

`__init__(self: rti.connextdds.ServiceRequest.DataWriterListener) → None`

`__module__ = 'rti.connextdds'`

`on_application_acknowledgment(self: rti.connextdds.ServiceRequest.DataWriterListener, arg0: rti.connextdds.ServiceRequest.DataWriter, arg1: rti.connextdds.AcknowledgmentInfo) → None`

On application acknowledgment callback

`on_instance_replaced(self: rti.connextdds.ServiceRequest.DataWriterListener, arg0: rti.connextdds.ServiceRequest.DataWriter, arg1: rti.connextdds.InstanceHandle) → None`

On instance replaced callback.

`on_liveliness_lost(self: rti.connextdds.ServiceRequest.DataWriterListener, arg0: rti.connextdds.ServiceRequest.DataWriter, arg1: rti.connextdds.LivelinessLostStatus) → None`

Liveliness lost callback.

`on_offered_deadline_missed(self: rti.connextdds.ServiceRequest.DataWriterListener, arg0: rti.connextdds.ServiceRequest.DataWriter, arg1: rti.connextdds.OfferedDeadlineMissedStatus) → None`

Offered deadline missed callback.

`on_offered_incompatible_qos(self: rti.connextdds.ServiceRequest.DataWriterListener, arg0: rti.connextdds.ServiceRequest.DataWriter, arg1: rti.connextdds.OfferedIncompatibleQosStatus) → None`

Offered incompatible QoS callback.

`on_publication_matched(self: rti.connextdds.ServiceRequest.DataWriterListener, arg0: rti.connextdds.ServiceRequest.DataWriter, arg1: rti.connextdds.PublicationMatchedStatus) → None`

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connexdds.ServiceRequest.DataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.ReliableReaderActivity-ChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connexdds.ServiceRequest.DataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.ReliableWriter-CacheChangedStatus) → None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connexdds.ServiceRequest.DataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class DataWriterSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *arg0*: rti.connexdds.ServiceRequest.DataWriterSeq) → *rti.connexdds.ServiceRequest.DataWriterSeq*

__bool__ (*self*: rti.connexdds.ServiceRequest.DataWriterSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *x*: rti.connexdds.ServiceRequest.DataWriter) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *arg0*: rti.connexdds.ServiceRequest.DataWriterSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__(self: rti.connextdds.ServiceRequest.DataWriterSeq, s: slice) -> rti.connextdds.ServiceRequest.DataWriterSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: int) -> rti.connextdds.ServiceRequest.DataWriter`

`__hash__ = None`

`__iadd__` (*self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: rti.connextdds.ServiceRequest.DataWriterSeq*) → *rti.connextdds.ServiceRequest.DataWriterSeq*

`__imul__` (*self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: int*) → *rti.connextdds.ServiceRequest.DataWriterSeq*

`__init__` (**args, **kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.ServiceRequest.DataWriterSeq) -> None`

2. `__init__(self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: rti.connextdds.ServiceRequest.DataWriterSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: Iterable) -> None`

`__iter__` (*self: rti.connextdds.ServiceRequest.DataWriterSeq*) → *Iterator*

`__len__` (*self: rti.connextdds.ServiceRequest.DataWriterSeq*) → *int*

`__module__ = 'rti.connextdds'`

`__mul__` (*self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: int*) → *rti.connextdds.ServiceRequest.DataWriterSeq*

`__ne__` (*self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: rti.connextdds.ServiceRequest.DataWriterSeq*) → *bool*

`__rmul__` (*self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: int*) → *rti.connextdds.ServiceRequest.DataWriterSeq*

`__setitem__` (**args, **kwargs*)

Overloaded function.

1. `__setitem__(self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: int, arg1: rti.connextdds.ServiceRequest.DataWriter) -> None`

2. `__setitem__(self: rti.connextdds.ServiceRequest.DataWriterSeq, arg0: slice, arg1: rti.connextdds.ServiceRequest.DataWriterSeq) -> None`

Assign list elements using a slice object

`append` (*self: rti.connextdds.ServiceRequest.DataWriterSeq, x: rti.connextdds.ServiceRequest.DataWriter*) → *None*

Add an item to the end of the list

clear (*self*: rti.connexdds.ServiceRequest.DataWriterSeq) → None

Clear the contents

count (*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *x*:
rti.connexdds.ServiceRequest.DataWriter) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *L*: rti.connexdds.ServiceRequest.DataWriterSeq) -> None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *i*: int, *x*:
rti.connexdds.ServiceRequest.DataWriter) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.ServiceRequest.DataWriterSeq) -> rti.connexdds.ServiceRequest.DataWriter

Remove and return the last item

2. pop(*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *i*: int) -> rti.connexdds.ServiceRequest.DataWriter

Remove and return the item at index *i*

remove (*self*: rti.connexdds.ServiceRequest.DataWriterSeq, *x*:
rti.connexdds.ServiceRequest.DataWriter) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class ITopicDescription

Bases: *IEntity*

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

property name

The name of the entity conforming to the ITopicDescription interface.

property participant

The parent DomainParticipant.

property type_name

The name of the associated type.

class LoanedSample

Bases: pybind11_object

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connextdds.ServiceRequest.LoanedSample) -> None

Basic constructor

2. **__init__**(self: rti.connextdds.ServiceRequest.LoanedSample, data: rti.connextdds.ServiceRequest, info: rti.connextdds.SampleInfo) -> None

Construct LoanedSample with data and info.

__iter__ (self: rti.connextdds.ServiceRequest.LoanedSample) → object

__module__ = 'rti.connextdds'

property data

Get the data associated with the sample.

property info

Get the info associated with the sample.

class LoanedSamples

Bases: pybind11_object

__enter__ (self: rti.connextdds.ServiceRequest.LoanedSamples) → *rti.connextdds.ServiceRequest.LoanedSamples*

Enter a context for the loaned samples, loan returned on context exit.

__exit__ (self: rti.connextdds.ServiceRequest.LoanedSamples, arg0: object, arg1: object, arg2: object) → None

Exit the context for the loaned samples, returning the resources.

__getitem__ (self: rti.connextdds.ServiceRequest.LoanedSamples, arg0: int) → *rti.connextdds.ServiceRequest.LoanedSample*

Access a LoanedSample object in an array-like syntax

__init__ (self: rti.connextdds.ServiceRequest.LoanedSamples) → None

Create an empty LoanedSamples object.

__iter__ (self: rti.connextdds.ServiceRequest.LoanedSamples) → Iterator

__len__ (self: rti.connextdds.ServiceRequest.LoanedSamples) → int

Get the number of samples in the loan.

__module__ = 'rti.connextdds'

property length

Get the number of samples in the loan.

return_loan (self: rti.connextdds.ServiceRequest.LoanedSamples) → None

Returns the loan to the DataReader.

class NoOpDataReaderListener

Bases: *DataReaderListener*

`__init__` (*self*: rti.connexdds.ServiceRequest.NoOpDataReaderListener) → None

`__module__` = 'rti.connexdds'

`on_data_available` (*self*: rti.connexdds.ServiceRequest.NoOpDataReaderListener, *arg0*: rti.connexdds.ServiceRequest.DataReader) → None

Data available callback.

`on_liveliness_changed` (*self*: rti.connexdds.ServiceRequest.NoOpDataReaderListener, *arg0*: rti.connexdds.ServiceRequest.DataReader, *arg1*: rti.connexdds.LivelinessChangedStatus) → None

Liveliness changed callback.

`on_requested_deadline_missed` (*self*: rti.connexdds.ServiceRequest.NoOpDataReaderListener, *arg0*: rti.connexdds.ServiceRequest.DataReader, *arg1*: rti.connexdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

`on_requested_incompatible_qos` (*self*: rti.connexdds.ServiceRequest.NoOpDataReaderListener, *arg0*: rti.connexdds.ServiceRequest.DataReader, *arg1*: rti.connexdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

`on_sample_lost` (*self*: rti.connexdds.ServiceRequest.NoOpDataReaderListener, *arg0*: rti.connexdds.ServiceRequest.DataReader, *arg1*: rti.connexdds.SampleLostStatus) → None

Sample lost callback.

`on_sample_rejected` (*self*: rti.connexdds.ServiceRequest.NoOpDataReaderListener, *arg0*: rti.connexdds.ServiceRequest.DataReader, *arg1*: rti.connexdds.SampleRejectedStatus) → None

Sample rejected callback.

`on_subscription_matched` (*self*: rti.connexdds.ServiceRequest.NoOpDataReaderListener, *arg0*: rti.connexdds.ServiceRequest.DataReader, *arg1*: rti.connexdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class NoOpDataWriterListener

Bases: *DataWriterListener*

`__init__` (*self*: rti.connexdds.ServiceRequest.NoOpDataWriterListener) → None

`__module__` = 'rti.connexdds'

on_application_acknowledgment (*self*: rti.connexdds.ServiceRequest.NoOp-DataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.AcknowledgmentInfo) → None

On application acknowledgment callback

on_instance_replaced (*self*: rti.connexdds.ServiceRequest.NoOpDataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.InstanceHandle) → None

On instance replaced callback.

on_liveliness_lost (*self*: rti.connexdds.ServiceRequest.NoOpDataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.LivelinessLostStatus) → None

Liveliness lost callback.

on_offered_deadline_missed (*self*: rti.connexdds.ServiceRequest.NoOp-DataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

on_offered_incompatible_qos (*self*: rti.connexdds.ServiceRequest.NoOp-DataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

on_publication_matched (*self*: rti.connexdds.ServiceRequest.NoOpDataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.PublicationMatchedStatus) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connexdds.ServiceRequest.NoOpDataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.ReliableReaderActivity-ChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connexdds.ServiceRequest.NoOp-DataWriterListener, *arg0*: rti.connexdds.ServiceRequest.DataWriter, *arg1*: rti.connexdds.ReliableWriter-CacheChangedStatus) → None

Reliable writer cache changed callback.

```
on_service_request_accepted (self: rti.connextdds.ServiceRequest.NoOp-
    DataWriterListener, arg0:
    rti.connextdds.ServiceRequest.DataWriter, arg1:
    rti.connextdds.ServiceRequest.AcceptedStatus) →
    None
```

On service request accepted callback.

```
class NoOpTopicListener
```

Bases: *TopicListener*

```
__init__ (self: rti.connextdds.ServiceRequest.NoOpTopicListener) → None
```

```
__module__ = 'rti.connextdds'
```

```
on_inconsistent_topic (self: rti.connextdds.ServiceRequest.NoOpTopicListener, arg0:
    rti.connextdds.ServiceRequest.Topic, arg1:
    rti.connextdds.InconsistentTopicStatus) → None
```

Inconsistent topic callback.

```
class Sample
```

Bases: *pybind11_object*

```
__init__ (*args, **kwargs)
```

Overloaded function.

```
1. __init__(self: rti.connextdds.ServiceRequest.Sample, data: rti.connextdds.Ser-
    viceRequest, info: rti.connextdds.SampleInfo) -> None
```

Construct Sample with data and info.

```
2. __init__(self: rti.connextdds.ServiceRequest.Sample, sample: rti.connextdds.Ser-
    viceRequest.Sample) -> None
```

Copy constructor.

```
3. __init__(self: rti.connextdds.ServiceRequest.Sample) -> None
```

Basic constructor

```
4. __init__(self: rti.connextdds.ServiceRequest.Sample, loaned_sample: rti.con-
    nextdds.ServiceRequest.LoanedSample) -> None
```

Construct a sample with a loaned sample.

```
__iter__ (self: rti.connextdds.ServiceRequest.Sample) → object
```

```
__module__ = 'rti.connextdds'
```

```
property data
```

The data associated with the sample.

```
property info
```

Get the info associated with the sample.

```
class SharedSamples
```

Bases: *pybind11_object*

__getitem__ (*self*: rti.connextdds.ServiceRequest.SharedSamples, *arg0*: int) → *rti.connextdds.ServiceRequest.LoanedSample*

Get the sample at the specified index.

__init__ (*self*: rti.connextdds.ServiceRequest.SharedSamples, *loaned_samples*: rti.connextdds.ServiceRequest.LoanedSamples) → None

Constructs an instance of SharedSamples, removing ownership of the loan from the Loaned Samples.

__iter__ (*self*: rti.connextdds.ServiceRequest.SharedSamples) → Iterator

Make a sample iterator

__len__ (*self*: rti.connextdds.ServiceRequest.SharedSamples) → int

Returns the number of samples.

__module__ = 'rti.connextdds'

class Topic

Bases: *ITopicDescription, IAnyTopic*

__eq__ (*self*: rti.connextdds.ServiceRequest.Topic, *arg0*: rti.connextdds.ServiceRequest.Topic) → bool

Test for equality.

__hash__ = None

__init__ (**args, **kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.ServiceRequest.Topic, *entity*: rti.connextdds.IEntity) -> None

Cast an Entity to a Topic.

2. **__init__**(*self*: rti.connextdds.ServiceRequest.Topic, *topic_description*: rti.connextdds.ServiceRequest.ITopicDescription) -> None

Cast an ITopicDescription to a Topic.

3. **__init__**(*self*: rti.connextdds.ServiceRequest.Topic, *topic*: rti.connextdds.IAnyTopic) -> None

Create a typed Topic from an AnyTopic.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.ServiceRequest.Topic, *arg0*: rti.connextdds.ServiceRequest.Topic) → bool

Test for inequality.

static find (*participant*: rti.connextdds.DomainParticipant, *name*: str) → Optional[*rti.connextdds.ServiceRequest.Topic*]

Look up a Topic by its name in the DomainParticipant.

property inconsistent_topic_status

Get a copy of the current InconsistentTopicStatus for this Topic.

property listener

The listener.

property qos

Get the TopicQos for this Topic.

This property's getter returns a deep copy.

set_listener (*self*: rti.connextdds.ServiceRequest.Topic, *listener*: rti.connextdds.ServiceRequest.TopicListener, *event_mask*: rti.connextdds.StatusMask) → None

Set the listener and event mask.

class TopicDescription

Bases: *ITopicDescription*

__eq__ (*self*: rti.connextdds.ServiceRequest.TopicDescription, *arg0*: rti.connextdds.ServiceRequest.TopicDescription) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.ServiceRequest.TopicDescription, *topic*: rti.connextdds.ServiceRequest.ITopicDescription) → None

Cast a Topic to a TopicDescription.

2. **__init__**(*self*: rti.connextdds.ServiceRequest.TopicDescription, *entity*: rti.connextdds.IEntity) → None

Cast a Topic to a TopicDescription.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.ServiceRequest.TopicDescription, *arg0*: rti.connextdds.ServiceRequest.TopicDescription) → bool

Test for inequality.

class TopicListener

Bases: *pybind11_object*

__init__ (*self*: rti.connextdds.ServiceRequest.TopicListener) → None

__module__ = 'rti.connextdds'

on_inconsistent_topic (*self*: rti.connextdds.ServiceRequest.TopicListener, *arg0*: rti.connextdds.ServiceRequest.Topic, *arg1*: rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class TopicSeq

Bases: *pybind11_object*

__add__ (*self*: rti.connexdds.ServiceRequest.TopicSeq, *arg0*: rti.connexdds.ServiceRequest.TopicSeq) → *rti.connexdds.ServiceRequest.TopicSeq*

__bool__ (*self*: rti.connexdds.ServiceRequest.TopicSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.ServiceRequest.TopicSeq, *x*: rti.connexdds.ServiceRequest.Topic) → bool
Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
Overloaded function.
1. **__delitem__**(*self*: rti.connexdds.ServiceRequest.TopicSeq, *arg0*: int) -> None
Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connexdds.ServiceRequest.TopicSeq, *arg0*: slice) -> None
Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.ServiceRequest.TopicSeq, *arg0*: rti.connexdds.ServiceRequest.TopicSeq) → bool

__getitem__ (**args*, ***kwargs*)
Overloaded function.
1. **__getitem__**(*self*: rti.connexdds.ServiceRequest.TopicSeq, *s*: slice) -> rti.connexdds.ServiceRequest.TopicSeq
Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.ServiceRequest.TopicSeq, *arg0*: int) -> rti.connexdds.ServiceRequest.Topic

__hash__ = None

__iadd__ (*self*: rti.connexdds.ServiceRequest.TopicSeq, *arg0*: rti.connexdds.ServiceRequest.TopicSeq) → *rti.connexdds.ServiceRequest.TopicSeq*

__imul__ (*self*: rti.connexdds.ServiceRequest.TopicSeq, *arg0*: int) → *rti.connexdds.ServiceRequest.TopicSeq*

__init__ (**args*, ***kwargs*)
Overloaded function.
1. **__init__**(*self*: rti.connexdds.ServiceRequest.TopicSeq) -> None
2. **__init__**(*self*: rti.connexdds.ServiceRequest.TopicSeq, *arg0*: rti.connexdds.ServiceRequest.TopicSeq) -> None
Copy constructor
3. **__init__**(*self*: rti.connexdds.ServiceRequest.TopicSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.ServiceRequest.TopicSeq) → Iterator

__len__ (*self*: rti.connexdds.ServiceRequest.TopicSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connextdds.ServiceRequest.TopicSeq, *arg0*: int) →
rti.connextdds.ServiceRequest.TopicSeq

__ne__ (*self*: rti.connextdds.ServiceRequest.TopicSeq, *arg0*:
rti.connextdds.ServiceRequest.TopicSeq) → bool

__rmul__ (*self*: rti.connextdds.ServiceRequest.TopicSeq, *arg0*: int) →
rti.connextdds.ServiceRequest.TopicSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connextdds.ServiceRequest.TopicSeq, *arg0*: int, *arg1*: rti.connextdds.ServiceRequest.Topic) -> None
2. **__setitem__**(*self*: rti.connextdds.ServiceRequest.TopicSeq, *arg0*: slice, *arg1*: rti.connextdds.ServiceRequest.TopicSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connextdds.ServiceRequest.TopicSeq, *x*:
rti.connextdds.ServiceRequest.Topic) → None

Add an item to the end of the list

clear (*self*: rti.connextdds.ServiceRequest.TopicSeq) → None

Clear the contents

count (*self*: rti.connextdds.ServiceRequest.TopicSeq, *x*: rti.connextdds.ServiceRequest.Topic)
 → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connextdds.ServiceRequest.TopicSeq, L: rti.connextdds.ServiceRequest.TopicSeq) -> None
2. **extend**(*self*: rti.connextdds.ServiceRequest.TopicSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.ServiceRequest.TopicSeq, *i*: int, *x*:
rti.connextdds.ServiceRequest.Topic) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connextdds.ServiceRequest.TopicSeq) -> rti.connextdds.ServiceRequest.Topic
2. **pop**(*self*: rti.connextdds.ServiceRequest.TopicSeq, *i*: int) -> rti.connextdds.ServiceRequest.Topic

Remove and return the last item

Remove and return the item at index *i*

remove (*self*: rti.connextdds.ServiceRequest.TopicSeq, *x*:
rti.connextdds.ServiceRequest.Topic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class ValidLoanedSamples

Bases: pybind11_object

__enter__ (*self*: rti.connextdds.ServiceRequest.ValidLoanedSamples) →
rti.connextdds.ServiceRequest.ValidLoanedSamples

__exit__ (*self*: rti.connextdds.ServiceRequest.ValidLoanedSamples, *arg0*: object, *arg1*: object,
arg2: object) → None

__init__ (**args*, ***kwargs*)

__iter__ (*self*: rti.connextdds.ServiceRequest.ValidLoanedSamples) → Iterator

__module__ = 'rti.connextdds'

class WriterContentFilter

Bases: ContentFilter

__init__ (*self*: rti.connextdds.ServiceRequest.WriterContentFilter) → None

__module__ = 'rti.connextdds'

writer_attach (*self*: rti.connextdds.ServiceRequest.WriterContentFilter) →
Optional[object]

A writer-side filtering API to create some state that can facilitate filtering on the writer side.

writer_compile (*self*: rti.connextdds.ServiceRequest.WriterContentFilter,
writer_filter_data: Optional[object], *property*:
rti.connextdds.ExpressionProperty, *expression*: str, *parameters*:
rti.connextdds.StringSeq, *type_code*:
Optional[rti.connextdds.DynamicType], *type_class_name*: str, *cookie*:
rti.connextdds.Cookie) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

writer_detach (*self*: rti.connextdds.ServiceRequest.WriterContentFilter, *writer_filter_data*:
Optional[object]) → None

A writer-side filtering API to clean up a previously created state using `writer_attach`.

writer_evaluate (*self*: rti.connextdds.ServiceRequest.WriterContentFilter,
writer_filter_data: Optional[object], *sample*:
rti.connextdds.ServiceRequest, *meta_data*:
rti.connextdds.FilterSampleInfo) → *rti.connextdds.CookieVector*

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

writer_finalize (*self*: rti.connexdds.ServiceRequest.WriterContentFilter, *writer_filter_data*: *Optional[object]*, *cookie*: rti.connexdds.Cookie) → None

A writer-side filtering API to clean up a previously compiled instance of the content filter.

writer_return_loan (*self*: rti.connexdds.ServiceRequest.WriterContentFilter, *writer_filter_data*: *Optional[object]*, *cookies*: rti.connexdds.CookieVector) → None

A writer-side filtering API to return the loan on the list of DataReaders returned by `writer_evaluate`.

class WriterContentFilterHelper

Bases: *WriterContentFilter*

__init__ (*self*: rti.connexdds.ServiceRequest.WriterContentFilterHelper) → None

__module__ = 'rti.connexdds'

add_cookie (*self*: rti.connexdds.ServiceRequest.WriterContentFilterHelper, *cookie*: rti.connexdds.Cookie) → None

A helper function which will add a Cookie to the Cookie sequence that is then returned by the `writer_evaluate` function.

writer_evaluate_helper (*self*: rti.connexdds.ServiceRequest.WriterContentFilterHelper, *writer_filter_data*: *Optional[object]*, *sample*: rti.connexdds.ServiceRequest, *meta_data*: rti.connexdds.FilterSampleInfo) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

__eq__ (*self*: rti.connexdds.ServiceRequest, *arg0*: rti.connexdds.ServiceRequest) → bool
Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ServiceRequest, *arg0*: rti.connexdds.ServiceRequest) → bool
Test for inequality.

property instance_id

Get the instance id of the request.

property request_body

Get the request body of the request.

property service_id

The service id of the request.


```
topic_name = 'DDSServiceRequest'
```

```
class rti.connexdds.ServiceRequestAcceptedStatus
```

```
Bases: pybind11_object
```

Information about the service_request_accepted status. Currently, the only service that causes the ServiceRequestAcceptedStatus to be triggered is TopicQuery service. A ServiceRequest is accepted when a DataWriter matches with a DataReader that has created a TopicQuery. This status is also changed (and the listener, if any, called) when a ServiceRequest has been cancelled, or deleted. This will happen when a DataReader deletes a TopicQuery using TopicQuery.close().

```
__init__ (*args, **kwargs)
```

```
__module__ = 'rti.connexdds'
```

```
property current_count
```

The current number of ServiceRequests that have been accepted by this DataWriter.

```
property last_request_handle
```

A handle to the last ServiceRequest that caused the DataWriter's status to change.

```
property service_id
```

ID of the service to which the accepted Request belongs.

```
property total_count
```

The total number of ServiceRequests that have been accepted by this DataWriter.

```
class rti.connexdds.ServiceRequestId
```

```
Bases: pybind11_object
```

```
LOCATOR_REACHABILITY = <ServiceRequestId.LOCATOR_REACHABILITY: 2>
```

```
MONITORING_LIBRARY_COMMAND =  
<ServiceRequestId.MONITORING_LIBRARY_COMMAND: 4>
```

```
MONITORING_LIBRARY_REPLY =  
<ServiceRequestId.MONITORING_LIBRARY_REPLY: 5>
```

```
class ServiceRequestId
```

```
Bases: pybind11_object
```

Members:

UNKNOWN : An unknown service.

TOPIC_QUERY : The topic query service.

LOCATOR_REACHABILITY : The locator reachability service.

MONITORING_LIBRARY_COMMAND : The Monitoring Library 2.0 command service.

MONITORING_LIBRARY_REPLY : The Monitoring Library 2.0 reply service.

```

LOCATOR_REACHABILITY = <ServiceRequestId.LOCATOR_REACHABILITY:
2>

MONITORING_LIBRARY_COMMAND =
<ServiceRequestId.MONITORING_LIBRARY_COMMAND: 4>

MONITORING_LIBRARY_REPLY =
<ServiceRequestId.MONITORING_LIBRARY_REPLY: 5>

TOPIC_QUERY = <ServiceRequestId.TOPIC_QUERY: 1>

UNKNOWN = <ServiceRequestId.UNKNOWN: 0>

__eq__(self: object, other: object) → bool

__getstate__(self: object) → int

__hash__(self: object) → int

__index__(self: rti.connexdds.ServiceRequestId.ServiceRequestId) → int

__init__(self: rti.connexdds.ServiceRequestId.ServiceRequestId, value: int) → None

__int__(self: rti.connexdds.ServiceRequestId.ServiceRequestId) → int

__members__ = {'LOCATOR_REACHABILITY':
<ServiceRequestId.LOCATOR_REACHABILITY: 2>,
'MONITORING_LIBRARY_COMMAND':
<ServiceRequestId.MONITORING_LIBRARY_COMMAND: 4>,
'MONITORING_LIBRARY_REPLY':
<ServiceRequestId.MONITORING_LIBRARY_REPLY: 5>, 'TOPIC_QUERY':
<ServiceRequestId.TOPIC_QUERY: 1>, 'UNKNOWN':
<ServiceRequestId.UNKNOWN: 0>}

__module__ = 'rti.connexdds'

__ne__(self: object, other: object) → bool

__repr__(self: object) → str

__setstate__(self: rti.connexdds.ServiceRequestId.ServiceRequestId, state: int) → None

__str__()
    name(self: handle) -> str

property name

property value

TOPIC_QUERY = <ServiceRequestId.TOPIC_QUERY: 1>

UNKNOWN = <ServiceRequestId.UNKNOWN: 0>

```

__eq__ (*self*: rti.connexdds.ServiceRequestId, *arg0*: rti.connexdds.ServiceRequestId) → bool
Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.ServiceRequestId, *arg0*: rti.connexdds.ServiceRequestId) → bool
Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.ServiceRequestId, *arg0*: rti.connexdds.ServiceRequestId) → bool
Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)
Overloaded function.

- __init__**(*self*: rti.connexdds.ServiceRequestId) -> None
Initializes enum to 0.
- __init__**(*self*: rti.connexdds.ServiceRequestId, *arg0*: rti.connexdds.ServiceRequestId.ServiceRequestId) -> None
Copy constructor.

__int__ (*self*: rti.connexdds.ServiceRequestId) → *rti.connexdds.ServiceRequestId.ServiceRequestId*
Int conversion.

__le__ (*self*: rti.connexdds.ServiceRequestId, *arg0*: rti.connexdds.ServiceRequestId) → bool
Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.ServiceRequestId, *arg0*: rti.connexdds.ServiceRequestId) → bool
Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ServiceRequestId, *arg0*: rti.connexdds.ServiceRequestId) → bool
Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.ServiceRequestId) → str
String conversion.

property underlying
Retrieves the actual enumerated value.

class rti.connexdds.**ServiceRequestSeq**
Bases: pybind11_object

__add__ (*self*: rti.connexdds.ServiceRequestSeq, *arg0*: rti.connexdds.ServiceRequestSeq) → *rti.connexdds.ServiceRequestSeq*

__bool__ (*self*: rti.connexdds.ServiceRequestSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.ServiceRequestSeq, *x*: rti.connexdds.ServiceRequest) → bool
Return true the container contains x

`__delitem__` (**args, **kwargs*)

Overloaded function.

1. `__delitem__(self: rti.connexdds.ServiceRequestSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connexdds.ServiceRequestSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (*self: rti.connexdds.ServiceRequestSeq, arg0: rti.connexdds.ServiceRequestSeq*) → bool

`__getitem__` (**args, **kwargs*)

Overloaded function.

1. `__getitem__(self: rti.connexdds.ServiceRequestSeq, s: slice) -> rti.connexdds.ServiceRequestSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.ServiceRequestSeq, arg0: int) -> rti.connexdds.ServiceRequest`

`__hash__` = None

`__iadd__` (*self: rti.connexdds.ServiceRequestSeq, arg0: rti.connexdds.ServiceRequestSeq*) → *rti.connexdds.ServiceRequestSeq*

`__imul__` (*self: rti.connexdds.ServiceRequestSeq, arg0: int*) → *rti.connexdds.ServiceRequestSeq*

`__init__` (**args, **kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.ServiceRequestSeq) -> None`
2. `__init__(self: rti.connexdds.ServiceRequestSeq, arg0: rti.connexdds.ServiceRequestSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.ServiceRequestSeq, arg0: Iterable) -> None`

`__iter__` (*self: rti.connexdds.ServiceRequestSeq*) → Iterator

`__len__` (*self: rti.connexdds.ServiceRequestSeq*) → int

`__module__` = `'rti.connexdds'`

`__mul__` (*self: rti.connexdds.ServiceRequestSeq, arg0: int*) → *rti.connexdds.ServiceRequestSeq*

`__ne__` (*self: rti.connexdds.ServiceRequestSeq, arg0: rti.connexdds.ServiceRequestSeq*) → bool

`__rmul__` (*self: rti.connexdds.ServiceRequestSeq, arg0: int*) → *rti.connexdds.ServiceRequestSeq*

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.ServiceRequestSeq, arg0: int, arg1: rti.connextdds.ServiceRequest) -> None`
2. `__setitem__(self: rti.connextdds.ServiceRequestSeq, arg0: slice, arg1: rti.connextdds.ServiceRequestSeq) -> None`

Assign list elements using a slice object

append (self: rti.connextdds.ServiceRequestSeq, x: rti.connextdds.ServiceRequest) → None

Add an item to the end of the list

clear (self: rti.connextdds.ServiceRequestSeq) → None

Clear the contents

count (self: rti.connextdds.ServiceRequestSeq, x: rti.connextdds.ServiceRequest) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connextdds.ServiceRequestSeq, L: rti.connextdds.ServiceRequestSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.ServiceRequestSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.ServiceRequestSeq, i: int, x: rti.connextdds.ServiceRequest) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.ServiceRequestSeq) -> rti.connextdds.ServiceRequest`

Remove and return the last item

2. `pop(self: rti.connextdds.ServiceRequestSeq, i: int) -> rti.connextdds.ServiceRequest`

Remove and return the item at index i

remove (self: rti.connextdds.ServiceRequestSeq, x: rti.connextdds.ServiceRequest) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class rti.connextdds.ServiceRequestTimestampedSeq

Bases: pybind11_object

__add__ (self: rti.connextdds.ServiceRequestTimestampedSeq, arg0: rti.connextdds.ServiceRequestTimestampedSeq) → rti.connextdds.ServiceRequestTimestampedSeq

__bool__ (*self*: rti.connexdds.ServiceRequestTimestampedSeq) → bool
 Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.ServiceRequestTimestampedSeq, *x*:
Tuple[rti.connexdds.ServiceRequest, rti.connexdds.Time]) → bool
 Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*: int) -> None
 Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*: slice) -> None
 Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*:
 rti.connexdds.ServiceRequestTimestampedSeq) → bool

__getitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.ServiceRequestTimestampedSeq, *s*: slice) -> rti.connexdds.ServiceRequestTimestampedSeq
 Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*: int) -> *Tuple*[rti.connexdds.ServiceRequest, rti.connexdds.Time]

__hash__ = None

__iadd__ (*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*:
 rti.connexdds.ServiceRequestTimestampedSeq) →
rti.connexdds.ServiceRequestTimestampedSeq

__imul__ (*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*: int) →
rti.connexdds.ServiceRequestTimestampedSeq

__init__ (**args*, ***kwargs*)
 Overloaded function.

1. **__init__**(*self*: rti.connexdds.ServiceRequestTimestampedSeq) -> None
2. **__init__**(*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*: rti.connexdds.ServiceRequestTimestampedSeq) -> None
 Copy constructor
3. **__init__**(*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.ServiceRequestTimestampedSeq) → Iterator

__len__ (*self*: rti.connexdds.ServiceRequestTimestampedSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*: int) →
rti.connexdds.ServiceRequestTimestampedSeq

__ne__ (*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*:
rti.connexdds.ServiceRequestTimestampedSeq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule
object NULL>

__rmul__ (*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*: int) →
rti.connexdds.ServiceRequestTimestampedSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*: int, *arg1*: Tuple[rti.connexdds.ServiceRequest, rti.connexdds.Time]) -> None
2. **__setitem__**(*self*: rti.connexdds.ServiceRequestTimestampedSeq, *arg0*: slice, *arg1*: rti.connexdds.ServiceRequestTimestampedSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.ServiceRequestTimestampedSeq, *x*:
Tuple[rti.connexdds.ServiceRequest, rti.connexdds.Time]) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.ServiceRequestTimestampedSeq) → None

Clear the contents

count (*self*: rti.connexdds.ServiceRequestTimestampedSeq, *x*:
Tuple[rti.connexdds.ServiceRequest, rti.connexdds.Time]) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.ServiceRequestTimestampedSeq, *L*: rti.connexdds.ServiceRequestTimestampedSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.ServiceRequestTimestampedSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.ServiceRequestTimestampedSeq, *i*: int, *x*:
Tuple[rti.connexdds.ServiceRequest, rti.connexdds.Time]) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connexdds.ServiceRequestTimestampedSeq) -> Tuple[rti.connexdds.ServiceRequest, rti.connexdds.Time]

Remove and return the last item

2. pop(self: rti.connexdds.ServiceRequestTimestampedSeq, i: int) -> Tuple[rti.connexdds.ServiceRequest, rti.connexdds.Time]

Remove and return the item at index i

remove (self: rti.connexdds.ServiceRequestTimestampedSeq, x: Tuple[rti.connexdds.ServiceRequest, rti.connexdds.Time]) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

rti.connexdds.**ShortSeq**

alias of *Int16Seq*

rti.connexdds.**ShortType**

alias of *Int16Type*

class rti.connexdds.**StatusCondition**

Bases: *ICCondition*

__eq__ (self: rti.connexdds.StatusCondition, arg0: rti.connexdds.StatusCondition) → bool
Compare StatusCondition objects for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. __init__(self: rti.connexdds.StatusCondition, entity: rti.connexdds.IEntity) -> None

Obtain a reference to an entity's StatusCondition object

2. __init__(self: rti.connexdds.StatusCondition, condition: rti.connexdds.ICCondition) -> None

Downcast a Condition to a StatusCondition.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.StatusCondition, arg0: rti.connexdds.StatusCondition) → bool
Compare StatusCondition objects for inequality.

dispatch (self: rti.connexdds.StatusCondition) → None
Dispatches the functions registered with the condition.

property enabled_statuses

Get/set the enabled statuses for this condition.

property entity

Get the Entity associated with this StatusCondition.

reset_handler (*self*: rti.connexdds.StatusCondition) → None

Resets the handler for this StatusCondition.

set_handler (*self*: rti.connexdds.StatusCondition, *func*:
Callable[[rti.connexdds.StatusCondition], None]) → None

Set a handler function for this StatusCondition.

property trigger_value

The trigger value of the condition.

class rti.connexdds.StatusMask

Bases: pybind11_object

A set of statuses.

ALL = 11111111111111111111111111111111

DATAREADER_CACHE = 00010000000000000000000000000000

DATAREADER_PROTOCOL = 00100000000000000000000000000000

DATAWRITER_APPLICATION_ACKNOWLEDGMENT =
00000000010000000000000000000000

DATAWRITER_CACHE = 00000100000000000000000000000000

DATAWRITER_INSTANCE_REPLACED = 00000000100000000000000000000000

DATAWRITER_PROTOCOL = 00001000000000000000000000000000

DATA_AVAILABLE = 00000000000000000000010000000000

DATA_ON_READERS = 00000000000000000000010000000000

DESTINATION_UNREACHABLE = 01000000000000000000000000000000

INCONSISTENT_TOPIC = 00000000000000000000000000000001

INVALID_LOCAL_IDENTITY_ADVANCE_NOTICE =
00000000000100000000000000000000

LIVELINESS_CHANGED = 00000000000000000001000000000000

LIVELINESS_LOST = 00000000000000000001000000000000

NONE = 00000000000000000000000000000000

OFFERED_DEADLINE_MISSED = 00000000000000000000000000000010

OFFERED_INCOMPATIBLE_QOS = 00000000000000000000000001000000

```

PUBLICATION_MATCHED = 000000000000000001000000000000
RELIABLE_READER_ACTIVITY_CHANGED = 000001000000000000000000000000
RELIABLE_WRITER_CACHE_CHANGED = 000000100000000000000000000000
REQUESTED_DEADLINE_MISSED = 00000000000000000000000000000100
REQUESTED_INCOMPATIBLE_QOS = 00000000000000000000000001000000
SAMPLE_LOST = 00000000000000000000000001000000
SAMPLE_REJECTED = 00000000000000000000000001000000
SAMPLE_REMOVED = 10000000000000000000000000000000
SERVICE_REQUEST_ACCEPTED = 00000000010000000000000000000000
SUBSCRIPTION_MATCHED = 00000000000000000100000000000000

```

`__and__` (*self*: rti.connexdds.StatusMask, *arg0*: rti.connexdds.StatusMask) → *rti.connexdds.StatusMask*

Bitwise logical AND of masks.

`__bool__` (*self*: rti.connexdds.StatusMask) → bool

Test if any bits are set.

`__contains__` (*self*: rti.connexdds.StatusMask, *arg0*: rti.connexdds.StatusMask) → bool

`__eq__` (*self*: rti.connexdds.StatusMask, *arg0*: rti.connexdds.StatusMask) → bool

Compare masks for equality.

`__getitem__` (*self*: rti.connexdds.StatusMask, *arg0*: int) → bool

Get individual mask bit.

`__hash__` = None

`__iand__` (*self*: rti.connexdds.StatusMask, *arg0*: rti.connexdds.StatusMask) → *rti.connexdds.StatusMask*

Set mask to logical AND with another mask.

`__ilshift__` (*self*: rti.connexdds.StatusMask, *arg0*: int) → *rti.connexdds.StatusMask*

Left shift bits in mask.

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.StatusMask) -> None`

Create a StatusMask equivalent to StatusMask.NONE

2. `__init__(self: rti.connexdds.StatusMask, value: int) -> None`

Creates a mask from the bits in an integer.

__int__ (*self*: rti.connexdds.StatusMask) → int
Convert mask to int.

__ior__ (*self*: rti.connexdds.StatusMask, *arg0*: rti.connexdds.StatusMask) → *rti.connexdds.StatusMask*
Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.StatusMask, *arg0*: int) → *rti.connexdds.StatusMask*
Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.StatusMask, *arg0*: rti.connexdds.StatusMask) → *rti.connexdds.StatusMask*
Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.StatusMask, *arg0*: int) → *rti.connexdds.StatusMask*
Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.StatusMask, *arg0*: rti.connexdds.StatusMask) → bool
Compare masks for inequality.

__or__ (*self*: rti.connexdds.StatusMask, *arg0*: rti.connexdds.StatusMask) → *rti.connexdds.StatusMask*
Bitwise logical OR of masks.

__repr__ (*self*: rti.connexdds.StatusMask) → str
Convert mask to string.

__rshift__ (*self*: rti.connexdds.StatusMask, *arg0*: int) → *rti.connexdds.StatusMask*
Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.StatusMask, *arg0*: int, *arg1*: bool) → None
Set individual mask bit

__str__ (*self*: rti.connexdds.StatusMask) → str
Convert mask to string.

__xor__ (*self*: rti.connexdds.StatusMask, *arg0*: rti.connexdds.StatusMask) → *rti.connexdds.StatusMask*
Bitwise logical XOR of masks.

property count
Returns the number of bits set in the mask.

flip (*args, **kwargs)
Overloaded function.

- flip(*self*: rti.connexdds.StatusMask) -> rti.connexdds.StatusMask
Flip all bits in the mask.
- flip(*self*: rti.connexdds.StatusMask, pos: int) -> rti.connexdds.StatusMask

Flip the mask bit at the specified position.

reset (*args, **kwargs)

Overloaded function.

1. reset(self: rti.connexdds.StatusMask) -> rti.connexdds.StatusMask

Clear all bits in the mask.

2. reset(self: rti.connexdds.StatusMask, pos: int) -> rti.connexdds.StatusMask

Clear the mask bit at the specified position.

set (*args, **kwargs)

Overloaded function.

1. set(self: rti.connexdds.StatusMask) -> rti.connexdds.StatusMask

Set all bits in the mask.

2. set(self: rti.connexdds.StatusMask, pos: int, value: bool = True) -> rti.connexdds.StatusMask

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (self: rti.connexdds.StatusMask, pos: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (self: rti.connexdds.StatusMask) → bool

Test if all bits are set.

test_any (self: rti.connexdds.StatusMask) → bool

Test if any bits are set.

test_none (self: rti.connexdds.StatusMask) → bool

Test if none of the bits are set.

class rti.connexdds.StreamKind

Bases: pybind11_object

ANY = <rti.connexdds.StreamKind object>

LIVE = <rti.connexdds.StreamKind object>

TOPIC_QUERY = <rti.connexdds.StreamKind object>

__and__ (self: rti.connexdds.StreamKind, arg0: rti.connexdds.StreamKind) → rti.connexdds.StreamKind

Bitwise logical AND of masks.

__bool__ (self: rti.connexdds.StreamKind) → rti.connexdds.StreamKind

Test if any bits are set.

__contains__ (*self*: rti.connexdds.StreamKind, *arg0*: rti.connexdds.StreamKind) → bool

__eq__ (*self*: rti.connexdds.StreamKind, *arg0*: rti.connexdds.StreamKind) → bool

Compare masks for equality.

__getitem__ (*self*: rti.connexdds.StreamKind, *arg0*: int) → bool

Get individual mask bit.

__hash__ = None

__iand__ (*self*: rti.connexdds.StreamKind, *arg0*: rti.connexdds.StreamKind) →
rti.connexdds.StreamKind

Set mask to logical AND with another mask.

__ilshift__ (*self*: rti.connexdds.StreamKind, *arg0*: int) → *rti.connexdds.StreamKind*

Left shift bits in mask.

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.StreamKind) -> None

Create a StreamKind with nothing enabled.

2. **__init__**(*self*: rti.connexdds.StreamKind, value: int) -> None

Creates a mask from the bits in an integer.

__int__ (*self*: rti.connexdds.StreamKind) → int

Convert mask to int.

__ior__ (*self*: rti.connexdds.StreamKind, *arg0*: rti.connexdds.StreamKind) →
rti.connexdds.StreamKind

Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.StreamKind, *arg0*: int) → *rti.connexdds.StreamKind*

Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.StreamKind, *arg0*: rti.connexdds.StreamKind) →
rti.connexdds.StreamKind

Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.StreamKind, *arg0*: int) → *rti.connexdds.StreamKind*

Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.StreamKind, *arg0*: rti.connexdds.StreamKind) → bool

Compare masks for inequality.

__or__ (*self*: rti.connexdds.StreamKind, *arg0*: rti.connexdds.StreamKind) →
rti.connexdds.StreamKind

Bitwise logical OR of masks.

__rshift__ (*self*: rti.connexdds.StreamKind, *arg0*: int) → rti.connexdds.StreamKind

Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.StreamKind, *arg0*: int, *arg1*: bool) → None

Set individual mask bit

__str__ (*self*: rti.connexdds.StreamKind) → str

__xor__ (*self*: rti.connexdds.StreamKind, *arg0*: rti.connexdds.StreamKind) → rti.connexdds.StreamKind

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (*args, **kwargs)

Overloaded function.

1. flip(*self*: rti.connexdds.StreamKind) -> rti.connexdds.StreamKind

Flip all bits in the mask.

2. flip(*self*: rti.connexdds.StreamKind, *pos*: int) -> rti.connexdds.StreamKind

Flip the mask bit at the specified position.

reset (*args, **kwargs)

Overloaded function.

1. reset(*self*: rti.connexdds.StreamKind) -> rti.connexdds.StreamKind

Clear all bits in the mask.

2. reset(*self*: rti.connexdds.StreamKind, *pos*: int) -> rti.connexdds.StreamKind

Clear the mask bit at the specified position.

set (*args, **kwargs)

Overloaded function.

1. set(*self*: rti.connexdds.StreamKind) -> rti.connexdds.StreamKind

Set all bits in the mask.

2. set(*self*: rti.connexdds.StreamKind, *pos*: int, *value*: bool = True) -> rti.connexdds.StreamKind

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (*self*: rti.connexdds.StreamKind, *pos*: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (*self*: rti.connexdds.StreamKind) → bool

Test if all bits are set.

test_any (*self*: rti.connexdds.StreamKind) → bool

Test if any bits are set.

test_none (*self*: rti.connexdds.StreamKind) → bool

Test if none of the bits are set.

class rti.connexdds.**StringMap**

Bases: pybind11_object

__bool__ (*self*: rti.connexdds.StringMap) → bool

Check whether the map is nonempty

__contains__ (**args*, ***kwargs*)

Overloaded function.

1. **__contains__**(*self*: rti.connexdds.StringMap, *arg0*: str) -> bool

2. **__contains__**(*self*: rti.connexdds.StringMap, *arg0*: object) -> bool

__delitem__ (*self*: rti.connexdds.StringMap, *arg0*: str) → None

__getitem__ (*self*: rti.connexdds.StringMap, *arg0*: str) → str

__init__ (*self*: rti.connexdds.StringMap) → None

__iter__ (*self*: rti.connexdds.StringMap) → Iterator

__len__ (*self*: rti.connexdds.StringMap) → int

__module__ = 'rti.connexdds'

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (*self*: rti.connexdds.StringMap) → str

Return the canonical string representation of this map.

__setitem__ (*self*: rti.connexdds.StringMap, *arg0*: str, *arg1*: str) → None

items (*self*: rti.connexdds.StringMap) → rti.connexdds.ItemsView[*StringMap*]

keys (*self*: rti.connexdds.StringMap) → rti.connexdds.KeysView[*StringMap*]

values (*self*: rti.connexdds.StringMap) → rti.connexdds.ValuesView[*StringMap*]

class rti.connexdds.**StringPairSeq**

Bases: pybind11_object

__add__ (*self*: rti.connexdds.StringPairSeq, *arg0*: rti.connexdds.StringPairSeq) →
rti.connexdds.StringPairSeq

__bool__ (*self*: rti.connexdds.StringPairSeq) → bool
 Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.StringPairSeq, *x*: Tuple[str, str]) → bool
 Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.StringPairSeq, *arg0*: int) -> None
 Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connexdds.StringPairSeq, *arg0*: slice) -> None
 Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.StringPairSeq, *arg0*: rti.connexdds.StringPairSeq) → bool

__getitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.StringPairSeq, *s*: slice) -> rti.connexdds.StringPairSeq
 Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.StringPairSeq, *arg0*: int) -> Tuple[str, str]

__hash__ = None

__iadd__ (*self*: rti.connexdds.StringPairSeq, *arg0*: rti.connexdds.StringPairSeq) → rti.connexdds.StringPairSeq

__imul__ (*self*: rti.connexdds.StringPairSeq, *arg0*: int) → rti.connexdds.StringPairSeq

__init__ (**args*, ***kwargs*)
 Overloaded function.

1. **__init__**(*self*: rti.connexdds.StringPairSeq) -> None
2. **__init__**(*self*: rti.connexdds.StringPairSeq, *arg0*: rti.connexdds.StringPairSeq) -> None
 Copy constructor
3. **__init__**(*self*: rti.connexdds.StringPairSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.StringPairSeq) → Iterator

__len__ (*self*: rti.connexdds.StringPairSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.StringPairSeq, *arg0*: int) → rti.connexdds.StringPairSeq

__ne__ (*self*: rti.connexdds.StringPairSeq, *arg0*: rti.connexdds.StringPairSeq) → bool

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>`

`__rmul__ (self: rti.connextdds.StringPairSeq, arg0: int) → rti.connextdds.StringPairSeq`

`__setitem__ (*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connextdds.StringPairSeq, arg0: int, arg1: Tuple[str, str]) -> None`
2. `__setitem__(self: rti.connextdds.StringPairSeq, arg0: slice, arg1: rti.connextdds.StringPairSeq) -> None`

Assign list elements using a slice object

`append (self: rti.connextdds.StringPairSeq, x: Tuple[str, str]) → None`

Add an item to the end of the list

`clear (self: rti.connextdds.StringPairSeq) → None`

Clear the contents

`count (self: rti.connextdds.StringPairSeq, x: Tuple[str, str]) → int`

Return the number of times x appears in the list

`extend (*args, **kwargs)`

Overloaded function.

1. `extend(self: rti.connextdds.StringPairSeq, L: rti.connextdds.StringPairSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.StringPairSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

`insert (self: rti.connextdds.StringPairSeq, i: int, x: Tuple[str, str]) → None`

Insert an item at a given position.

`pop (*args, **kwargs)`

Overloaded function.

1. `pop(self: rti.connextdds.StringPairSeq) -> Tuple[str, str]`

Remove and return the last item

2. `pop(self: rti.connextdds.StringPairSeq, i: int) -> Tuple[str, str]`

Remove and return the item at index i

`remove (self: rti.connextdds.StringPairSeq, x: Tuple[str, str]) → None`

Remove the first item from the list whose value is x. It is an error if there is no such item.

`class rti.connextdds.StringSeq`

Bases: `pybind11_object`

__add__ (*self*: rti.connexdds.StringSeq, *arg0*: rti.connexdds.StringSeq) → *rti.connexdds.StringSeq*

__bool__ (*self*: rti.connexdds.StringSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.StringSeq, *x*: *str*) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.StringSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.StringSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.StringSeq, *arg0*: rti.connexdds.StringSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.StringSeq, *s*: slice) -> rti.connexdds.StringSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.StringSeq, *arg0*: int) -> str

__hash__ = None

__iadd__ (*self*: rti.connexdds.StringSeq, *arg0*: rti.connexdds.StringSeq) →
rti.connexdds.StringSeq

__imul__ (*self*: rti.connexdds.StringSeq, *arg0*: int) → *rti.connexdds.StringSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.StringSeq) -> None

2. **__init__**(*self*: rti.connexdds.StringSeq, *arg0*: rti.connexdds.StringSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.StringSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.StringSeq) → Iterator

__len__ (*self*: rti.connexdds.StringSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.StringSeq, *arg0*: int) → *rti.connexdds.StringSeq*

__ne__ (*self*: rti.connexdds.StringSeq, *arg0*: rti.connexdds.StringSeq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (*self*: rti.connexdds.StringSeq) → str

Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.StringSeq, *arg0*: int) → rti.connexdds.StringSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.StringSeq, *arg0*: int, *arg1*: str) -> None

2. **__setitem__**(*self*: rti.connexdds.StringSeq, *arg0*: slice, *arg1*: rti.connexdds.StringSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.StringSeq, *x*: str) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.StringSeq) → None

Clear the contents

count (*self*: rti.connexdds.StringSeq, *x*: str) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.StringSeq, *L*: rti.connexdds.StringSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.StringSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.StringSeq, *i*: int, *x*: str) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.StringSeq) -> str

Remove and return the last item

2. **pop**(*self*: rti.connexdds.StringSeq, *i*: int) -> str

Remove and return the item at index *i*

remove (*self*: rti.connexdds.StringSeq, *x*: str) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class `rti.connexdds.StringType`Bases: *UnidimensionalCollectionType*`__eq__` (*self*: `rti.connexdds.StringType`, *arg0*: `rti.connexdds.StringType`) → bool

Test for equality.

`__hash__` = None`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.StringType, bounds: int) -> None`

Creates a bounded string.

2. `__init__(self: rti.connexdds.StringType) -> None`

Creates an unbounded string.

`__module__` = `'rti.connexdds'``__ne__` (*self*: `rti.connexdds.StringType`, *arg0*: `rti.connexdds.StringType`) → bool

Test for inequality.

class `rti.connexdds.StructType`Bases: *ACTMember*`__eq__` (*self*: `rti.connexdds.StructType`, *arg0*: `rti.connexdds.StructType`) → bool

Test for equality.

`__hash__` = None`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.StructType, name: str) -> None`

Creates an empty StructType.

2. `__init__(self: rti.connexdds.StructType, name: str, extensibility: rti.connexdds.ExtensibilityKind) -> None`

Creates an empty StructType with a specified extensibility.

3. `__init__(self: rti.connexdds.StructType, name: str, parent: rti.connexdds.StructType) -> None`

Create an empty StructType with a base type.

4. `__init__(self: rti.connexdds.StructType, name: str, parent: rti.connexdds.StructType, extensibility: rti.connexdds.ExtensibilityKind) -> None`

Creates an empty StructType with a base type and specified extensibility.

5. `__init__(self: rti.connexdds.StructType, name: str, members: rti.connexdds.MemberSeq) -> None`

Create a StructType with the provided members.

6. `__init__(self: rti.connextdds.StructType, name: str, members: rti.connextdds.MemberSeq, extensibility: rti.connextdds.ExtensibilityKind) -> None`

Creates a StructType with the provided members and specified extensibility.

7. `__init__(self: rti.connextdds.StructType, name: str, parent: rti.connextdds.StructType, members: rti.connextdds.MemberSeq) -> None`

Create a StructType with a base type and the provided members.

8. `__init__(self: rti.connextdds.StructType, name: str, parent: rti.connextdds.StructType, members: rti.connextdds.MemberSeq, extensibility: rti.connextdds.ExtensibilityKind) -> None`

Creates a StructType with a base types, the provided members, and the specified extensibility.

9. `__init__(self: rti.connextdds.StructType, type: rti.connextdds.DynamicType) -> None`

Cast a DynamicType to a StructType

```
__module__ = 'rti.connextdds'
```

```
__ne__ (self: rti.connextdds.StructType, arg0: rti.connextdds.StructType) → bool
```

Test for inequality.

```
add_member (self: rti.connextdds.StructType, member: rti.connextdds.Member) →  
    rti.connextdds.StructType
```

Adds a member at the end.

```
add_members (self: rti.connextdds.StructType, members: rti.connextdds.MemberSeq) →  
    rti.connextdds.StructType
```

Adds all members of the sequence to the end.

```
property extensibility_kind
```

Struct's extensibility kind.

```
find_member_by_id (self: rti.connextdds.StructType, member_id: int) → int
```

Get the index of the member with a specific ID.

```
property has_parent
```

Indicates if this type has a base type.

```
property parent
```

Retrieve the base type.

```
class rti.connextdds.Subscriber
```

```
Bases:  IEntity
```

```
__eq__ (self: rti.connextdds.Subscriber, arg0: rti.connextdds.Subscriber) → bool
```

Test for equality.

```
__hash__ = None
```

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.Subscriber, participant: rti.connextdds.DomainParticipant) -> None`

Create a subscriber under a DomainParticipant.

2. `__init__(self: rti.connextdds.Subscriber, participant: rti.connextdds.DomainParticipant, qos: rti.connextdds.SubscriberQos, listener: rti.connextdds.SubscriberListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Create a Subscriber under a DomainParticipant with a listener.

3. `__init__(self: rti.connextdds.Subscriber, entity: rti.connextdds.IEntity) -> None`

Cast an Entity to a Subscriber.

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.Subscriber, arg0: rti.connextdds.Subscriber) → bool

Test for inequality.

property default_datareader_qos

The default DataReaderQos.

This property's getter returns a deep copy.

find_datareader (self: rti.connextdds.Subscriber, name: str) → Optional[rti.connextdds.AnyDataReader]

Find a DataReader in this Subscriber by its name.

find_datareader_by_topic_name (self: rti.connextdds.Subscriber, topic_name: str) → Optional[rti.connextdds.AnyDataReader]

Find a DataReader in this Subscriber by its topic name. If more than one exists for this Topic, the first one found is returned.

find_datareaders (*args, **kwargs)

Overloaded function.

1. `find_datareaders(self: rti.connextdds.Subscriber) -> rti.connextdds.AnyDataReaderSeq`

Find all DataReaders in the Subscriber.

2. `find_datareaders(self: rti.connextdds.Subscriber, arg0: rti.connextdds.DataState) -> rti.connextdds.AnyDataReaderSeq`

Find all DataReaders that contain samples of the given DataState in the Subscriber.

3. `find_datareaders(self: rti.connextdds.Subscriber, arg0: str) -> rti.connextdds.AnyDataReaderSeq`

Find all DataReaders for a given topic name

property listener

Get the listener.

notify_datareaders (*self*: rti.connextdds.Subscriber) → None

This operation invokes the operation on_data_available on the DataReaderListener objects attached to contained DataReader entities with a DATA_AVAILABLE status that is considered changed

property participant

Get the parent DomainParticipant for this Subscriber.

property qos

The SubscriberQos for this Subscriber.

This property's getter returns a deep copy.

set_listener (*self*: rti.connextdds.Subscriber, *listener*: rti.connextdds.SubscriberListener, *event_mask*: rti.connextdds.StatusMask) → None

Bind the listener and event mask to the Subscriber.

class rti.connextdds.SubscriberListener

Bases: AnyDataReaderListener

__init__ (*self*: rti.connextdds.SubscriberListener) → None

__module__ = 'rti.connextdds'

on_data_on_readers (*self*: rti.connextdds.SubscriberListener, *arg0*: rti.connextdds.Subscriber) → None

Data on datareaders callback.

class rti.connextdds.SubscriberQos

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.SubscriberQos, *arg0*: rti.connextdds.SubscriberQos) → bool

Test for equality

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.SubscriberQos) -> None

Create a SubscriberQos with the default value for each policy.

2. **__init__**(*self*: rti.connextdds.SubscriberQos, *subscriber*: rti.connextdds.Subscriber) -> None

Create a SubscriberQos with settings equivalent to those of the provided Subscriber object.

3. **__init__**(*self*: rti.connextdds.SubscriberQos, *other*: rti.connextdds.SubscriberQos) -> None

Create a copy of a SubscriberQos object.

__lshift__ (*args, **kwargs)

Overloaded function.

1. `__lshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.Presentation) -> rti.connextdds.SubscriberQos`

Set the PresentationQoS.

2. `__lshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.Partition) -> rti.connextdds.SubscriberQos`

Set the PartitionQoS.

3. `__lshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.GroupData) -> rti.connextdds.SubscriberQos`

Set the GroupDataQoS.

4. `__lshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.EntityFactory) -> rti.connextdds.SubscriberQos`

Set the EntityFactoryQoS.

5. `__lshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.ExclusiveArea) -> rti.connextdds.SubscriberQos`

Set the ExclusiveAreaQoS.

6. `__lshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.EntityName) -> rti.connextdds.SubscriberQos`

Set the EntityNameQoS.

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.SubscriberQos) → bool

Test for inequality.

__rshift__ (*args, **kwargs)

Overloaded function.

1. `__rshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.Presentation) -> rti.connextdds.SubscriberQos`

Get the PresentationQoS.

2. `__rshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.Partition) -> rti.connextdds.SubscriberQos`

Get the PartitionQoS.

3. `__rshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.GroupData) -> rti.connextdds.SubscriberQos`

Get the GroupDataQoS.

4. `__rshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.EntityFactory) -> rti.connextdds.SubscriberQos`

Get the EntityFactoryQoS.

5. `__rshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.ExclusiveArea) -> rti.connextdds.SubscriberQos`

Get the ExclusiveAreaQoS.

6. `__rshift__(self: rti.connextdds.SubscriberQos, arg0: rti.connextdds.EntityName) -> rti.connextdds.SubscriberQos`

Get the EntityNameQoS.

`__str__ (self: rti.connextdds.SubscriberQos) → str`

property entity_factory

Get/set EntityFactory QoS.

property entity_name

Get/set EntityName QoS.

property exclusive_area

Get/set ExclusiveArea QoS.

property group_data

Get/set GroupData QoS.

property partition

Get/set Partition QoS.

property presentation

Get/set Presentation QoS.

to_string (*self: rti.connextdds.SubscriberQos, format: rti.connextdds.QosPrintFormat = QosPrintFormat(), base: Optional[rti.connextdds.SubscriberQos] = None, print_all: bool = False*) → str

Convert QoS to string based on params.

class `rti.connextdds.SubscriberSeq`

Bases: `pybind11_object`

`__add__ (self: rti.connextdds.SubscriberSeq, arg0: rti.connextdds.SubscriberSeq) → rti.connextdds.SubscriberSeq`

`__bool__ (self: rti.connextdds.SubscriberSeq) → bool`

Check whether the list is nonempty

`__contains__ (self: rti.connextdds.SubscriberSeq, x: rti.connextdds.Subscriber) → bool`

Return true the container contains x

`__delitem__ (*args, **kwargs)`

Overloaded function.

1. `__delitem__(self: rti.connextdds.SubscriberSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connextdds.SubscriberSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__(self: rti.connextdds.SubscriberSeq, arg0: rti.connextdds.SubscriberSeq) → bool`

`__getitem__(*args, **kwargs)`

Overloaded function.

1. `__getitem__(self: rti.connextdds.SubscriberSeq, s: slice) -> rti.connextdds.SubscriberSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.SubscriberSeq, arg0: int) -> rti.connextdds.Subscriber`

`__hash__ = None`

`__iadd__(self: rti.connextdds.SubscriberSeq, arg0: rti.connextdds.SubscriberSeq) → rti.connextdds.SubscriberSeq`

`__imul__(self: rti.connextdds.SubscriberSeq, arg0: int) → rti.connextdds.SubscriberSeq`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.SubscriberSeq) -> None`
2. `__init__(self: rti.connextdds.SubscriberSeq, arg0: rti.connextdds.SubscriberSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.SubscriberSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.SubscriberSeq) → Iterator`

`__len__(self: rti.connextdds.SubscriberSeq) → int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.SubscriberSeq, arg0: int) → rti.connextdds.SubscriberSeq`

`__ne__(self: rti.connextdds.SubscriberSeq, arg0: rti.connextdds.SubscriberSeq) → bool`

`__rmul__(self: rti.connextdds.SubscriberSeq, arg0: int) → rti.connextdds.SubscriberSeq`

`__setitem__(*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connextdds.SubscriberSeq, arg0: int, arg1: rti.connextdds.Subscriber) -> None`
2. `__setitem__(self: rti.connextdds.SubscriberSeq, arg0: slice, arg1: rti.connextdds.SubscriberSeq) -> None`

Assign list elements using a slice object

append (*self*: rti.connexdds.SubscriberSeq, *x*: rti.connexdds.Subscriber) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.SubscriberSeq) → None

Clear the contents

count (*self*: rti.connexdds.SubscriberSeq, *x*: rti.connexdds.Subscriber) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connexdds.SubscriberSeq, *L*: rti.connexdds.SubscriberSeq) -> None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connexdds.SubscriberSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.SubscriberSeq, *i*: int, *x*: rti.connexdds.Subscriber) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.SubscriberSeq) -> rti.connexdds.Subscriber

Remove and return the last item

2. pop(*self*: rti.connexdds.SubscriberSeq, *i*: int) -> rti.connexdds.Subscriber

Remove and return the item at index *i*

remove (*self*: rti.connexdds.SubscriberSeq, *x*: rti.connexdds.Subscriber) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.SubscriptionBuiltinTopicData

Bases: pybind11_object

class ContentFilter

Bases: ContentFilterBase

__init__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilter) → None

__module__ = 'rti.connexdds'

compile (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilter, *expression*: str, *parameters*: rti.connexdds.StringSeq, *type_code*: Optional[rti.connexdds.DynamicType], *type_class_name*: str, *old_compile_data*: Optional[object]) → Optional[object]

Compile an instance of the content filter according to the filter expression and parameters of the given data type.

evaluate (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilter, *compile_data*: *Optional*[object], *sample*: rti.connexdds.SubscriptionBuiltinTopicData, *meta_data*: rti.connexdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

finalize (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilter, *compile_data*: *Optional*[object]) → None

A previously compiled instance of the content filter is no longer in use and resources can now be cleaned up.

class ContentFilteredTopic

Bases: *ITopicDescription*, *IAnyTopic*

__eq__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *topic*: rti.connexdds.SubscriptionBuiltinTopicData.Topic, *name*: str, *contentfilter*: rti.connexdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. **__init__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *topic_description*: rti.connexdds.SubscriptionBuiltinTopicData.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → bool

Test for inequality.

append_to_expression_parameter (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *index*: int, *extension*: str) → None

Append the extension to the end of parameter at the provided index, separated by a comma.

property filter_expression

Get the filter expression

property filter_parameters

Get/set the filter parameters.

static find (*participant*: rti.connexdds.DomainParticipant, *name*: str) → *Optional*[rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

remove_from_expression_parameter (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *index*: int, *remove_term*: str) → None

Removes the specified term from the parameter at the provided index.

set_filter (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *arg0*: rti.connexdds.Filter) → None

Set the filter.

property topic

Get the underlying Topic.

class ContentFilteredTopicSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq

__bool__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) -> None

Delete the list elements at index i

2. **__delitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *s*: slice) -> rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) -> rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic

__hash__ = None

__iadd__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq

__imul__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) → rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → None
2. **__init__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → None

Copy constructor

3. **__init__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: Iterable) → None

__iter__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → Iterator

__len__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) → rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq

__ne__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → bool

__rmul__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) → rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int, *arg1*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → None
2. **__setitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: slice, *arg1*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → None

Assign list elements using a slice object

append (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → None

Clear the contents

count (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *L*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *L*: Iterable) → None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *i*: int, *x*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq) → rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic

Remove and return the last item

2. pop(*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *i*: int) → rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic

Remove and return the item at index *i*

remove (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class DataReader

Bases: *IDataReader*

class Selector

Bases: *pybind11_object*

__init__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector, *datareader*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader) → None

Create a Selector for a DataReader to read/take based on selected conditions

__module__ = 'rti.connexdds'

condition (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector, *condition*: rti.connexdds.IReadCondition) → *rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector*

Select samples based on a ReadCondition.

content (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector, *query*: rti.connexdds.Query) → *rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector*

Select samples based on a Query.

instance (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector,
handle: rti.connexdds.InstanceHandle) →
rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector

Select a specific instance to read/take.

max_samples (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector,
max: int) →
rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector

Limit the number of samples read/taken by the Select.

next_instance (*self*:
 rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector,
previous: rti.connexdds.InstanceHandle) →
rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector

Select the instance after the specified instance to read/take.

read (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector) → list
 Read copies of available samples (data and info) based on the Selector settings.

read_data (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector) →
 list

Read copies of available valid data based on the Selector settings.

read_loaned (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector)
 → *rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples*

Take available samples (data and info) based on the Selector settings and return them in a loaned container.

state (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector, *state*:
 rti.connexdds.DataState) →
rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector

Select samples with a specified data state.

take (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector) → list
 Take copies of available samples (data and info) based on the Selector settings.

take_data (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector) →
 list

Take copies of available valid data based on the Selector settings.

take_loaned (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector)
 → *rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples*

Read available samples (data and info) based on the Selector settings and return them in a loaned container.

__enter__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader) →
rti.connexdds.SubscriptionBuiltinTopicData.DataReader

Enter a context for this DataReader, to be cleaned up on exiting context

__eq__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → bool

Test for equality.

__exit__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *arg0*: object, *arg1*: object, *arg2*: object) → None

Exit the context for this DataReader, cleaning up resources.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *topic*: rti.connextdds.SubscriptionBuiltinTopicData.Topic) -> None

Create a DataReader in the implicit subscriber with default QoS.

2. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *topic*: rti.connextdds.SubscriptionBuiltinTopicData.Topic, *qos*: rti.connextdds.DataReaderQos, *listener*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener = None, *mask*: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader in the implicit subscriber with specific QoS and a listener.

3. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *cft*: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) -> None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with default QoS.

4. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *cft*: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *qos*: rti.connextdds.DataReaderQos, *listener*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener = None, *mask*: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with specific QoS.

5. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *subscriber*: rti.connextdds.Subscriber, *topic*: rti.connextdds.SubscriptionBuiltinTopicData.Topic) -> None

Create a DataReader.

6. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *subscriber*: rti.connextdds.Subscriber, *topic*: rti.connextdds.SubscriptionBuiltinTopicData.Topic, *qos*: rti.connextdds.DataReaderQos, *listener*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener = None, *mask*: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader in a subscriber with specific QoS and a listener.

7. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *subscriber*: rti.connextdds.Subscriber, *cft*: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic) -> None

Create a DataReader with a ContentFilteredTopic in a subscriber with default QoS.

8. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *subscriber*: rti.connextdds.Subscriber, *cft*: rti.connextdds.SubscriptionBuiltinTopicData.ContentFilteredTopic, *qos*: rti.connextdds.DataReaderQos, *listener*: rti.connextdds.Subscrip-

tionBuiltinTopicData.DataReaderListener = None, mask: rti.connexdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader with a ContentFilteredTopic in a subscriber with specific QoS.

9. `__init__(self: rti.connexdds.SubscriptionBuiltinTopicData.DataReader, reader: rti.connexdds.IAnyDataReader)` -> None

Get a typed DataReader from an AnyDataReader.

10. `__init__(self: rti.connexdds.SubscriptionBuiltinTopicData.DataReader, entity: rti.connexdds.IEntity)` -> None

Get a typed DataReader from an Entity.

`__lshift__` (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader, *arg0*: rti.connexdds.DataReaderQos) → *rti.connexdds.SubscriptionBuiltinTopicData.DataReader*

Set the DataReaderQos for this DataReader.

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader) → bool

Test for inequality.

`__rshift__` (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader, *arg0*: rti.connexdds.DataReaderQos) → *rti.connexdds.SubscriptionBuiltinTopicData.DataReader*

Get the DataReaderQos from this DataReader

`acknowledge_all` (*args, **kwargs)

Overloaded function.

1. `acknowledge_all(self: rti.connexdds.SubscriptionBuiltinTopicData.DataReader)` -> None

Acknowledge all previously accessed samples.

2. `acknowledge_all(self: rti.connexdds.SubscriptionBuiltinTopicData.DataReader, arg0: rti.connexdds.AckResponseData)` -> None

Acknowledge all previously accessed samples.

`acknowledge_sample` (*args, **kwargs)

Overloaded function.

1. `acknowledge_sample(self: rti.connexdds.SubscriptionBuiltinTopicData.DataReader, sample_info: rti.connexdds.SampleInfo)` -> None

Acknowledge a single sample.

2. `acknowledge_sample(self: rti.connexdds.SubscriptionBuiltinTopicData.DataReader, sample_info: rti.connexdds.SampleInfo, ack_response_data: rti.connexdds.AckResponseData)` -> None

Acknowledge a single sample with ack response data.

`close` (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataReader) → None

Close this DataReader.

`property datareader_cache_status`

Get the DataReaderCacheStatus for the DataReader.

property datareader_protocol_status

Get the DataReaderProtocolStatus for the DataReader.

property default_filter_state

Returns the filter state for the read/take operations.

static find_all_by_topic (*subscriber*: rti.connextdds.Subscriber, *topic_name*: str) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq*

Retrieve all DataReaders for the given topic name in the subscriber.

static find_by_name (**args*, ***kwargs*)

Overloaded function.

1. **find_by_name**(participant: rti.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.SubscriptionBuiltinTopicData.DataReader]

Find DataReader in DomainParticipant with the DataReader's name, returning the first found.

2. **find_by_name**(subscriber: rti.connextdds.Subscriber, name: str) -> Optional[rti.connextdds.SubscriptionBuiltinTopicData.DataReader]

Find DataReader in Subscriber with the DataReader's name, returning the first found.

static find_by_topic (*subscriber*: rti.connextdds.Subscriber, *name*: str) → Optional[*rti.connextdds.SubscriptionBuiltinTopicData.DataReader*]

Find DataReader in Subscriber with a topic name, returning the first found.

is_matched_publication_alive (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *arg0*: rti.connextdds.InstanceHandle) → bool

Check if a matched publication is alive.

key_value (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.SubscriptionBuiltinTopicData*

Retrieve the instance key that corresponds to an instance handle.

property listener

Gets or sets the listener with StatusMask.ALL

property liveliness_changed_status

Get the LivelinessChangedStatus for this DataReader.

lookup_instance (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *key_holder*: rti.connextdds.SubscriptionBuiltinTopicData) → *rti.connextdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

matched_publication_data (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.PublicationBuiltinTopicData*

Get the PublicationBuiltinTopicData for a publication matched to this DataReader.

matched_publication_datareader_protocol_status (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *publication_handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.DataReaderProtocolStatus*

Get the DataReaderProtocolStatus for the DataReader based on the matched publication handle.

matched_publication_participant_data (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a publication matched to this DataReader.

property matched_publications

Get a copy of the list of the currently matched publication handles.

property qos

The DataReaderQos for this DataReader.

This property's getter returns a deep copy.

read (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → list

Read copies of all available samples (data and info).

read_data (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → list

Read copies of all available valid data.

read_loaned (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → *rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples*

Read all available samples (data and info) and return them in a loaned container.

property requested_deadline_missed_status

Get the RequestedDeadlineMissed status for the DataReader

property requested_incompatible_qos_status

Get the RequestedIncompatibleQosStatus for the DataReader.

property sample_lost_status

Get the SampleLostStatus for the DataReader.

property sample_rejected_status

Get the SampleRejectedStatus for the DataReader.

select (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → *dds::sub::DataReader<dds::topic::TSubscriptionBuiltinTopicData<rti::topic::SubscriptionBuiltinTopicDataImpl>, rti::sub::DataReaderImpl>::Selector*

Get a Selector to perform complex data selections, such as per-instance selection, content, and status filtering.

set_listener (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *listener*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener, *event_mask*: rti.connextdds.StatusMask) → None

Set the listener and associated event mask.

property subscriber

Returns the parent Subscriber of the DataReader.

property subscription_matched_status

Get the SubscriptionMatchedStatus for the DataReader.

take (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → list

Take copies of all available samples (data and info).

take_data (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → list

Take copies of all available valid data.

take_loaned (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → *rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples*

Take all available samples (data and info) and return them in a loaned container.

property topic_description

Returns the TopicDescription associated with the DataReader.

property topic_name

Get the topic name associated with this DataReader.

property type_name

Get the type name associated with this DataReader.

wait_for_historical_data (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *max_wait*: rti.connextdds.Duration) → None

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind.

wait_for_historical_data_async (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *max_wait*: rti.connextdds.Duration) → object

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind. This call is awaitable and only for use with asyncio.

class DataReaderListener

Bases: pybind11_object

__init__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener) → None

```
__module__ = 'rti.connextdds'
```

```
on_data_available (self:
                    rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener,
                    arg0: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) →
                    None
```

Data available callback.

```
on_liveliness_changed (self: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener,
                        arg0:
                        rti.connextdds.SubscriptionBuiltinTopicData.DataReader, arg1:
                        rti.connextdds.LivelinessChangedStatus) → None
```

Liveliness changed callback.

```
on_requested_deadline_missed (self: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener,
                               arg0:
                               rti.connextdds.SubscriptionBuiltinTopicData.DataReader, arg1:
                               rti.connextdds.RequestedDeadlineMissedStatus) →
                               None
```

Requested deadline missed callback.

```
on_requested_incompatible_qos (self: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener,
                                arg0:
                                rti.connextdds.SubscriptionBuiltinTopicData.DataReader, arg1:
                                rti.connextdds.RequestedIncompatibleQosStatus)
                                → None
```

Requested incompatible QoS callback.

```
on_sample_lost (self: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener,
                 arg0: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, arg1:
                 rti.connextdds.SampleLostStatus) → None
```

Sample lost callback.

```
on_sample_rejected (self:
                    rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener,
                    arg0: rti.connextdds.SubscriptionBuiltinTopicData.DataReader,
                    arg1: rti.connextdds.SampleRejectedStatus) → None
```

Sample rejected callback.

```
on_subscription_matched (self: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderListener,
                           arg0:
                           rti.connextdds.SubscriptionBuiltinTopicData.DataReader,
                           arg1: rti.connextdds.SubscriptionMatchedStatus) → None
```

Subscription matched callback.

```
class DataReaderSeq
```

```
    Bases: pybind11_object
```

__add__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq*

__bool__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *x*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → bool
Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
Overloaded function.
1. **__delitem__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: int) -> None
Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: slice) -> None
Delete list elements using a slice object

__eq__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) → bool

__getitem__ (**args*, ***kwargs*)
Overloaded function.
1. **__getitem__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *s*: slice) -> rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq
Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: int) -> rti.connextdds.SubscriptionBuiltinTopicData.DataReader

__hash__ = None

__iadd__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq*

__imul__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: int) → *rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq*

__init__ (**args*, ***kwargs*)
Overloaded function.
1. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) -> None
2. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) -> None
Copy constructor
3. **__init__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) → Iterator

__len__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) → int

__module__ = 'rti.connextdds'

__mul__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: int) → rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq

__ne__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) → bool

__rmul__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: int) → rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: int, *arg1*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) -> None
2. **__setitem__**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *arg0*: slice, *arg1*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *x*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → None

Add an item to the end of the list

clear (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) → None

Clear the contents

count (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *x*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *L*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *i*: int, *x*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq) -> rti.connextdds.SubscriptionBuiltinTopicData.DataReader

Remove and return the last item

2. `pop(self: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, i: int) -> rti.connextdds.SubscriptionBuiltinTopicData.DataReader`

Remove and return the item at index `i`

remove (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataReaderSeq, *x*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → None

Remove the first item from the list whose value is `x`. It is an error if there is no such item.

class DataWriter

Bases: *IEntity, IAnyDataWriter*

__enter__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → *rti.connextdds.SubscriptionBuiltinTopicData.DataWriter*

Enter a context for this DataWriter, to be cleaned up on exiting context

__eq__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → bool

Test for equality.

__exit__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg0*: object, *arg1*: object, *arg2*: object) → None

Exit the context for this DataWriter, cleaning up resources.

__hash__ = None

__init__ (**args, **kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, topic: rti.connextdds.SubscriptionBuiltinTopicData.Topic) -> None`

Creates a DataWriter in the implicit publisher with default QoS.

2. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, topic: rti.connextdds.SubscriptionBuiltinTopicData.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Creates a DataWriter in the implicit publisher with specific QoS and optionally a listener.

3. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.SubscriptionBuiltinTopicData.Topic) -> None`

Creates a DataWriter in a publisher with default QoS.

4. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.SubscriptionBuiltinTopicData.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Creates a DataWriter in a publisher with specific QoS and optionally a listener.

5. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, writer: rti.connextdds.IAnyDataWriter) -> None`

Create a typed DataWriter from an AnyDataWriter.

6. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, entity: rti.connextdds.IEntity) -> None`

Create a typed DataWriter from an Entity.

__lshift__ (*args, **kwargs)

Overloaded function.

1. `__lshift__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg0: rti.connextdds.DataWriterQos) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

Sets the DataWriterQos.

2. `__lshift__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg0: Tuple[rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.Time]) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

Writes a paired sample with a timestamp.

3. `__lshift__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg0: Tuple[rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.InstanceHandle]) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

Writes a paired sample with an instance handle.

4. `__lshift__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg0: rti.connextdds.SubscriptionBuiltinTopicData.TimestampedSeq) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

Writes a sequence of pairs of samples with timestamps.

5. `__lshift__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg0: rti.connextdds.SubscriptionBuiltinTopicDataSeq) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

Writes a sequence of samples.

6. `__lshift__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg0: rti.connextdds.SubscriptionBuiltinTopicData) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

Writes a sample.

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg0: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → bool

Test for inequality.

__rshift__ (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg0: rti.connextdds.DataWriterQos) → rti.connextdds.SubscriptionBuiltinTopicData.DataWriter

Get the DataWriterQos.

assert_liveliness (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → None

Manually asserts the liveliness of the DataWriter.

close (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → None

Close this DataWriter.

create_data (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → rti.connextdds.SubscriptionBuiltinTopicData

Create data of the writer's associated type and initialize it.

property datawriter_cache_status

Get a copy of the cache status for this writer.

property datawriter_protocol_status

Get a copy of the protocol status for this writer.

dispose_instance (*args, **kwargs)

Overloaded function.

1. `dispose_instance(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

Dispose an instance.

2. `dispose_instance(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

Dispose an instance with a timestamp.

3. `dispose_instance(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> None`

Dispose an instance with params.

dispose_instance_async (*args, **kwargs)

Overloaded function.

1. `dispose_instance_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> object`

Dispose an instance.

2. `dispose_instance_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Dispose an instance with a timestamp.

3. `dispose_instance_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, key_holder: rti.connextdds.SubscriptionBuiltinTopicData) -> object`

Dispose the instance associated with key_holder.

4. `dispose_instance_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, key_holder: rti.connextdds.SubscriptionBuiltinTopicData, timestamp: rti.connextdds.Time) -> object`

Dispose the instance associated with key_holder using a timestamp

5. `dispose_instance_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> object`

Dispose an instance with params.

static find_all_by_topic (publisher: rti.connextdds.Publisher, topic_name: str) → rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq

Retrieve all DataWriters for the given topic name in the publisher.

static find_by_name (*args, **kwargs)

Overloaded function.

1. `find_by_name(participant: rti.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.SubscriptionBuiltinTopicData.DataWriter]`

Find DataWriter in DomainParticipant with the provided name, returning the first found.

2. `find_by_name(publisher: rti.connextdds.Publisher, name: str) -> Optional[rti.connextdds.SubscriptionBuiltinTopicData.DataWriter]`

Find DataWriter in Publisher with the DataReader's name, returning the first found.

static find_by_topic (*publisher*: rti.connextdds.Publisher, *name*: str) → Optional[rti.connextdds.SubscriptionBuiltinTopicData.DataWriter]

Find DataWriter in publisher with a topic name, returning the first found.

flush (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → None

Flushes the batch in progress in the context of the calling thread.

is_matched_subscription_active (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *arg0*: rti.connextdds.InstanceHandle) → bool

A boolean indicating whether or not the matched subscription is active.

is_sample_app_acknowledged (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *sample_id*: rti.connextdds.SampleIdentity) → bool

Indicates if a sample is considered application-acknowledged.

key_value (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *handle*: rti.connextdds.InstanceHandle) → rti.connextdds.SubscriptionBuiltinTopicData

Retrieve the instance key that corresponds to an instance handle.

property listener

Get the listener associated with the DataWriter or set the listener.

property liveliness_lost_status

Get a copy of the LivelinessLostStatus.

lookup_instance (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *key_holder*: rti.connextdds.SubscriptionBuiltinTopicData) → rti.connextdds.InstanceHandle

Retrieve the instance handle that corresponds to an instance key_holder

matched_subscription_data (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *handle*: rti.connextdds.InstanceHandle) → rti.connextdds.SubscriptionBuiltinTopicData

Get the SubscriptionBuiltinTopicData for a subscription matched to this DataWriter.

matched_subscription_datawriter_protocol_status (*args, **kwargs)

Overloaded function.

1. `matched_subscription_datawriter_protocol_status(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.DataWriterProtocolStatus`

Get a copy of the protocol status for this writer per a matched subscription handle.

2. `matched_subscription_datawriter_protocol_status(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, locator: rti.connextdds.Locator) -> rti.connextdds.DataWriterProtocolStatus`

Get a copy of the protocol status for this writer per a matched subscription locator.

matched_subscription_participant_data (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a subscription matched to this DataWriter.

property matched_subscriptions

Get a copy of the list of the currently matched subscription handles.

property matched_subscriptions_locators

The locators used to communicate with matched DataReaders.

property offered_deadline_missed_status

Get a copy of the OfferedDeadlineMissedStatus.

property offered_incompatible_qos_status

Get a copy of the OfferedIncompatibleQosStatus

property publication_matched_status

Get a copy of the PublicationMatchedStatus

property publisher

Get the Publisher that owns this DataWriter.

property qos

The DataWriterQos for this DataWriter. This property's getter returns a deep copy.

register_instance (**args*, ***kwargs*)

Overloaded function.

1. register_instance(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *key_holder*: rti.connextdds.SubscriptionBuiltinTopicData) → rti.connextdds.InstanceHandle

Informs RTI Connex that the application will be modifying a particular instance.

2. register_instance(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *key_holder*: rti.connextdds.SubscriptionBuiltinTopicData, *timestamp*: rti.connextdds.Time) → rti.connextdds.InstanceHandle

Informs RTI Connex that the application will be modifying a particular instance and specified the timestamp.

3. register_instance(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *key_holder*: rti.connextdds.SubscriptionBuiltinTopicData, *params*: rti.connextdds.WriteParams) → rti.connextdds.InstanceHandle

Registers instance with parameters.

property reliable_reader_activity_changed_status

Get a copy of the reliable reader activity changed status for this writer.

property reliable_writer_cache_changed_status

Get a copy of the reliable cache status for this writer.

property service_request_accepted_status

Get a copy of the service request accepted status for this writer.

set_listener (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *listener*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener, *event_mask*: rti.connextdds.StatusMask) → None

Set the listener and event mask for the DataWriter.

property topic

Get the Topic object associated with this DataWriter.

property topic_name

Get the topic name associated with this DataWriter.

property type_name

Get the type name for the topic object associated with this DataWriter.

unregister_instance (**args*, ***kwargs*)

Overloaded function.

1. `unregister_instance(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

Unregister an instance.

2. `unregister_instance(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

Unregister an instance with timestamp.

3. `unregister_instance(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> None`

Unregister an instance with parameters.

unregister_instance_async (**args*, ***kwargs*)

Overloaded function.

1. `unregister_instance_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> object`

Unregister an instance.

2. `unregister_instance_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Unregister an instance with timestamp.

3. `unregister_instance_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, key_holder: rti.connextdds.SubscriptionBuiltinTopicData) -> object`

Unregister the instance associated with `key_holder`.

4. `unregister_instance_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, key_holder: rti.connextdds.SubscriptionBuiltinTopicData, timestamp: rti.connextdds.Time) -> object`

Unregister the instance associate with `key_holder` using a timestamp.

5. `unregister_instance_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> object`

Unregister an instance with parameters.

wait_for_acknowledgments (*self*:
 rti.connextdds.SubscriptionBuiltinTopicData.DataWriter,
 max_wait: rti.connextdds.Duration) → None

Blocks the calling thread until all data written by a reliable DataWriter is acknowledged or until the timeout expires.

wait_for_asynchronous_publishing (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *max_wait*:
 rti.connextdds.Duration) → None

This operation blocks the calling thread (up to *max_wait*) until all data written by the asynchronous DataWriter is sent and acknowledged (if reliable) by all matched DataReader entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that *max_wait* elapsed before all the data was sent and/or acknowledged.

In other words, this guarantees that sending to best effort DataReader is complete in addition to what DataWriter.wait_for_acknowledgments() provides.

If the DataWriter does not have PublishMode kind set to PublishModeKind.ASYNCHRONOUS the operation will complete immediately

wait_for_asynchronous_publishing_async (*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter,
 max_wait: rti.connextdds.Duration)
 → object

This function is awaitable until either a timeout of *max_wait* or all data written by the asynchronous DataWriter is sent and acknowledged (if reliable) by all matched DataReader entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that *max_wait* elapsed before all the data was sent and/or acknowledged. This function works with asyncio.

In other words, this guarantees that sending to best effort DataReader is complete in addition to what DataWriter.wait_for_acknowledgments() provides.

If the DataWriter does not have PublishMode kind set to PublishModeKind.ASYNCHRONOUS the operation will complete immediately

write (**args*, ***kwargs*)

Overloaded function.

1. write(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *samples*: rti.connextdds.SubscriptionBuiltinTopicDataSeq) -> None

Write a sequence of samples.

2. write(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *samples*: rti.connextdds.SubscriptionBuiltinTopicDataSeq, *timestamp*: rti.connextdds.Time) -> None

Write a sequence of samples with a timestamp.

3. write(*self*: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, *sample*: rti.connextdds.SubscriptionBuiltinTopicData) -> None

Write a sample.

4. `write(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, sample: rti.connextdds.SubscriptionBuiltinTopicData, timestamp: rti.connextdds.Time) -> None`

Write a sample with a specified timestamp.

5. `write(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, sample: rti.connextdds.SubscriptionBuiltinTopicData, handle: rti.connextdds.InstanceHandle) -> None`

Write a sample with an instance handle.

6. `write(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, sample: rti.connextdds.SubscriptionBuiltinTopicData, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> None`

Write a sample with an instance handle and specified timestamp.

7. `write(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, sample: rti.connextdds.SubscriptionBuiltinTopicData, params: rti.connextdds.WriteParams) -> None`

Write with advanced parameters.

write_async (*args, **kwargs)

Overloaded function.

1. `write_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, sample: rti.connextdds.SubscriptionBuiltinTopicData) -> object`

Write a sample. This method is awaitable and is only for use with asyncio.

2. `write_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, sample: rti.connextdds.SubscriptionBuiltinTopicData, timestamp: rti.connextdds.Time) -> object`

Write a sample with a specified timestamp. This methods is awaitable and only for use with asyncio.

3. `write_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, sample: rti.connextdds.SubscriptionBuiltinTopicData, handle: rti.connextdds.InstanceHandle) -> object`

Write a sample with an instance handle. This method is awaitable and only for use with asyncio.

4. `write_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, sample: rti.connextdds.SubscriptionBuiltinTopicData, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Write a sample with an instance handle and specified timestamp. This method is awaitable and only for use with asyncio.

5. `write_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, samples: rti.connextdds.SubscriptionBuiltinTopicDataSeq) -> object`

Write a sequence of samples. This method is awaitable and only for use with asyncio.

6. `write_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, samples: rti.connextdds.SubscriptionBuiltinTopicDataSeq, timestamp: rti.connextdds.Time) -> object`

Write a sequence of samples with a timestamp. This method is awaitable and only for use with asyncio.

7. `write_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, samples: rti.connextdds.SubscriptionBuiltinTopicDataSeq, handles: rti.connextdds.InstanceHandleSeq) -> object`

Write a sequence of samples with their instance handles. This method is awaitable and only for use with asyncio.

8. `write_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, samples: rti.connextdds.SubscriptionBuiltinTopicDataSeq, handles: rti.connextdds.InstanceHandleSeq, timestamp: rti.connextdds.Time) -> object`

Write a sequence of samples with their instance handles and a timestamp. This method is awaitable and only for use with asyncio.

```
9. write_async(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, sample:
    rti.connextdds.SubscriptionBuiltinTopicData, params: rti.connextdds.WriteParams) ->
    object
```

Write with advanced parameters.

class DataWriterListener

Bases: pybind11_object

```
__init__ (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener) → None
```

```
__module__ = 'rti.connextdds'
```

```
on_application_acknowledgment (self: rti.connextdds.SubscriptionBuiltinTopic-
    Data.DataWriterListener, arg0:
    rti.connextdds.SubscriptionBuiltinTopic-
    Data.DataWriter, arg1:
    rti.connextdds.AcknowledgmentInfo) → None
```

On application acknowledgment callback

```
on_instance_replaced (self: rti.connextdds.SubscriptionBuiltinTopic-
    Data.DataWriterListener, arg0:
    rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg1:
    rti.connextdds.InstanceHandle) → None
```

On instance replaced callback.

```
on_liveliness_lost (self:
    rti.connextdds.SubscriptionBuiltinTopicData.DataWriterListener,
    arg0: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter,
    arg1: rti.connextdds.LivelinessLostStatus) → None
```

Liveliness lost callback.

```
on_offered_deadline_missed (self: rti.connextdds.SubscriptionBuiltinTopic-
    Data.DataWriterListener, arg0:
    rti.connextdds.SubscriptionBuiltinTopic-
    Data.DataWriter, arg1:
    rti.connextdds.OfferedDeadlineMissedStatus) → None
```

Offered deadline missed callback.

```
on_offered_incompatible_qos (self: rti.connextdds.SubscriptionBuiltinTopic-
    Data.DataWriterListener, arg0:
    rti.connextdds.SubscriptionBuiltinTopic-
    Data.DataWriter, arg1:
    rti.connextdds.OfferedIncompatibleQosStatus) →
    None
```

Offered incompatible QoS callback.

on_publication_matched (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.PublicationMatchedStatus) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class DataWriterSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq) → *rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq*

__bool__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq, *x*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriter) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. `__delitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) → bool`

`__getitem__(*args, **kwargs)`

Overloaded function.

1. `__getitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, s: slice) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: int) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter`

`__hash__ = None`

`__iadd__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) → rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq`

`__imul__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: int) → rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) -> None`
2. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) → Iterator`

`__len__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) → int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: int) → rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq`

`__ne__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) → bool`

`__rmul__(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: int) → rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq`

__setitem__ (*args, **kwargs)

Overloaded function.

1. **__setitem__**(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: int, arg1: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) -> None

2. **__setitem__**(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, arg0: slice, arg1: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) -> None

Assign list elements using a slice object

append (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, x: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → None

Add an item to the end of the list

clear (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) → None

Clear the contents

count (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, x: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. **extend**(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, L: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, i: int, x: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. **pop**(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter

Remove and return the last item

2. **pop**(self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, i: int) -> rti.connextdds.SubscriptionBuiltinTopicData.DataWriter

Remove and return the item at index i

remove (self: rti.connextdds.SubscriptionBuiltinTopicData.DataWriterSeq, x: rti.connextdds.SubscriptionBuiltinTopicData.DataWriter) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class ITopicDescription

Bases: *IEntity*

__init__ (*args, **kwargs)

```
__module__ = 'rti.connexdds'
```

property name

The name of the entity conforming to the ITopicDescription interface.

property participant

The parent DomainParticipant.

property type_name

The name of the associated type.

class LoanedSample

Bases: pybind11_object

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connexdds.SubscriptionBuiltinTopicData.LoanedSample) -> None`

Basic constructor

2. `__init__(self: rti.connexdds.SubscriptionBuiltinTopicData.LoanedSample, data: rti.connexdds.SubscriptionBuiltinTopicData, info: rti.connexdds.SampleInfo) -> None`

Construct LoanedSample with data and info.

```
__iter__ (self: rti.connexdds.SubscriptionBuiltinTopicData.LoanedSample) → object
```

```
__module__ = 'rti.connexdds'
```

property data

Get the data associated with the sample.

property info

Get the info associated with the sample.

class LoanedSamples

Bases: pybind11_object

```
__enter__ (self: rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples) → rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples
```

Enter a context for the loaned samples, loan returned on context exit.

```
__exit__ (self: rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples, arg0: object, arg1: object, arg2: object) → None
```

Exit the context for the loaned samples, returning the resources.

```
__getitem__ (self: rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples, arg0: int) → rti.connexdds.SubscriptionBuiltinTopicData.LoanedSample
```

Access a LoanedSample object in an array-like syntax

```
__init__ (self: rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples) → None
```

Create an empty LoanedSamples object.

```
__iter__ (self: rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples) → Iterator
```

__len__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples) → int

Get the number of samples in the loan.

__module__ = 'rti.connextdds'

property length

Get the number of samples in the loan.

return_loan (*self*: rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples) → None

Returns the loan to the DataReader.

class NoOpDataReaderListener

Bases: *DataReaderListener*

__init__ (*self*: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener) → None

__module__ = 'rti.connextdds'

on_data_available (*self*: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader) → None

Data available callback.

on_liveliness_changed (*self*: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *arg1*: rti.connextdds.LivelinessChangedStatus) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *arg1*: rti.connextdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *arg1*: rti.connextdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connextdds.SubscriptionBuiltinTopicData.DataReader, *arg1*: rti.connextdds.SampleLostStatus) → None

Sample lost callback.

```
on_sample_rejected (self: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener, arg0:
                    rti.connextdds.SubscriptionBuiltinTopicData.DataReader, arg1:
                    rti.connextdds.SampleRejectedStatus) → None
```

Sample rejected callback.

```
on_subscription_matched (self: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener, arg0:
                          rti.connextdds.SubscriptionBuiltinTopicData.DataReader,
                          arg1: rti.connextdds.SubscriptionMatchedStatus) → None
```

Subscription matched callback.

```
class NoOpDataWriterListener
```

```
Bases: DataWriterListener
```

```
__init__ (self: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener) →
          None
```

```
__module__ = 'rti.connextdds'
```

```
on_application_acknowledgment (self: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, arg0:
                                rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg1:
                                rti.connextdds.AcknowledgmentInfo) → None
```

On application acknowledgment callback

```
on_instance_replaced (self: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, arg0:
                       rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg1:
                       rti.connextdds.InstanceHandle) → None
```

On instance replaced callback.

```
on_liveliness_lost (self: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, arg0:
                    rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg1:
                    rti.connextdds.LivelinessLostStatus) → None
```

Liveliness lost callback.

```
on_offered_deadline_missed (self: rti.connextdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, arg0:
                              rti.connextdds.SubscriptionBuiltinTopicData.DataWriter, arg1:
                              rti.connextdds.OfferedDeadlineMissedStatus) → None
```

Offered deadline missed callback.

on_offered_incompatible_qos (*self*: rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

on_publication_matched (*self*: rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.PublicationMatchedStatus) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class NoOpTopicListener

Bases: *TopicListener*

__init__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.NoOpTopicListener) → None

__module__ = 'rti.connexdds'

on_inconsistent_topic (*self*: rti.connexdds.SubscriptionBuiltinTopicData.NoOpTopicListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.Topic, *arg1*: rti.connexdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class Sample

Bases: pybind11_object

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connextdds.SubscriptionBuiltinTopicData.Sample, data: rti.connextdds.SubscriptionBuiltinTopicData, info: rti.connextdds.SampleInfo) -> None

Construct Sample with data and info.

2. **__init__**(self: rti.connextdds.SubscriptionBuiltinTopicData.Sample, sample: rti.connextdds.SubscriptionBuiltinTopicData.Sample) -> None

Copy constructor.

3. **__init__**(self: rti.connextdds.SubscriptionBuiltinTopicData.Sample) -> None

Basic constructor

4. **__init__**(self: rti.connextdds.SubscriptionBuiltinTopicData.Sample, loaned_sample: rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample) -> None

Construct a sample with a loaned sample.

__iter__ (self: rti.connextdds.SubscriptionBuiltinTopicData.Sample) → object

__module__ = 'rti.connextdds'

property data

The data associated with the sample.

property info

Get the info associated with the sample.

class SharedSamples

Bases: pybind11_object

__getitem__ (self: rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples, arg0: int) → *rti.connextdds.SubscriptionBuiltinTopicData.LoanedSample*

Get the sample at the specified index.

__init__ (self: rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples, loaned_samples: rti.connextdds.SubscriptionBuiltinTopicData.LoanedSamples) → None

Constructs an instance of SharedSamples, removing ownership of the loan from the Loaned Samples.

__iter__ (self: rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples) → Iterator

Make a sample iterator

__len__ (self: rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples) → int

Returns the number of samples.

__module__ = 'rti.connextdds'

class TopicBases: *ITopicDescription, IAnyTopic***__eq__** (*self*: rti.connexdds.SubscriptionBuiltinTopicData.Topic, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.Topic) → bool

Test for equality.

__hash__ = None**__init__** (**args, **kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.Topic, *entity*: rti.connexdds.IEntity) -> None

Cast an Entity to a Topic.

2. **__init__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.Topic, *topic_description*: rti.connexdds.SubscriptionBuiltinTopicData.ITopicDescription) -> None

Cast an ITopicDescription to a Topic.

3. **__init__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.Topic, *topic*: rti.connexdds.IAnyTopic) -> None

Create a typed Topic from an AnyTopic.

__module__ = 'rti.connexdds'**__ne__** (*self*: rti.connexdds.SubscriptionBuiltinTopicData.Topic, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.Topic) → bool

Test for inequality.

static find (*participant*: rti.connexdds.DomainParticipant, *name*: str) → Optional[rti.connexdds.SubscriptionBuiltinTopicData.Topic]

Look up a Topic by its name in the DomainParticipant.

property inconsistent_topic_status

Get a copy of the current InconsistentTopicStatus for this Topic.

property listener

The listener.

property qos

Get the TopicQos for this Topic.

This property's getter returns a deep copy.

set_listener (*self*: rti.connexdds.SubscriptionBuiltinTopicData.Topic, *listener*: rti.connexdds.SubscriptionBuiltinTopicData.TopicListener, *event_mask*: rti.connexdds.StatusMask) → None

Set the listener and event mask.

class TopicDescriptionBases: *ITopicDescription*

__eq__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription, *topic*: rti.connexdds.SubscriptionBuiltinTopicData.ITopicDescription) -> None

Cast a Topic to a TopicDescription.

2. **__init__**(*self*: rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription, *entity*: rti.connexdds.IEntity) -> None

Cast a Topic to a TopicDescription.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription) → bool

Test for inequality.

class TopicListener

Bases: pybind11_object

__init__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.TopicListener) → None

__module__ = 'rti.connexdds'

on_inconsistent_topic (*self*: rti.connexdds.SubscriptionBuiltinTopicData.TopicListener, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.Topic, *arg1*: rti.connexdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class TopicSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq) → rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq

__bool__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq, *x*: rti.connexdds.SubscriptionBuiltinTopicData.Topic) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. `__delitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: int) -> None`
Delete the list elements at index `i`

2. `__delitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: slice) -> None`
Delete list elements using a slice object

`__eq__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) -> bool`

`__getitem__(*args, **kwargs)`
Overloaded function.

1. `__getitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, s: slice) -> rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq`
Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: int) -> rti.connextdds.SubscriptionBuiltinTopicData.Topic`

`__hash__ = None`

`__iadd__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) -> rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq`

`__imul__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: int) -> rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq`

`__init__(*args, **kwargs)`
Overloaded function.

1. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) -> None`

2. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) -> None`
Copy constructor

3. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) -> Iterator`

`__len__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) -> int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: int) -> rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq`

`__ne__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) -> bool`

`__rmul__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: int) -> rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq`

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: int, arg1: rti.connextdds.SubscriptionBuiltinTopicData.Topic) -> None`
2. `__setitem__(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, arg0: slice, arg1: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) -> None`

Assign list elements using a slice object

append (self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, x: rti.connextdds.SubscriptionBuiltinTopicData.Topic) → None

Add an item to the end of the list

clear (self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) → None

Clear the contents

count (self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, x: rti.connextdds.SubscriptionBuiltinTopicData.Topic) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, L: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, i: int, x: rti.connextdds.SubscriptionBuiltinTopicData.Topic) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq) -> rti.connextdds.SubscriptionBuiltinTopicData.Topic`

Remove and return the last item

2. `pop(self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, i: int) -> rti.connextdds.SubscriptionBuiltinTopicData.Topic`

Remove and return the item at index i

remove (self: rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq, x: rti.connextdds.SubscriptionBuiltinTopicData.Topic) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class ValidLoanedSamples

Bases: pybind11_object

__enter__ (self: rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples) → rti.connextdds.SubscriptionBuiltinTopicData.ValidLoanedSamples

__exit__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ValidLoanedSamples, *arg0*: object, *arg1*: object, *arg2*: object) → None

__init__ (*args, **kwargs)

__iter__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.ValidLoanedSamples) → Iterator

__module__ = 'rti.connexdds'

class WriterContentFilter

Bases: *ContentFilter*

__init__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter) → None

__module__ = 'rti.connexdds'

writer_attach (*self*: rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter) → Optional[object]

A writer-side filtering API to create some state that can facilitate filtering on the writer side.

writer_compile (*self*: rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object], *property*: rti.connexdds.ExpressionProperty, *expression*: str, *parameters*: rti.connexdds.StringSeq, *type_code*: Optional[rti.connexdds.DynamicType], *type_class_name*: str, *cookie*: rti.connexdds.Cookie) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

writer_detach (*self*: rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object]) → None

A writer-side filtering API to clean up a previously created state using `writer_attach`.

writer_evaluate (*self*: rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object], *sample*: rti.connexdds.SubscriptionBuiltinTopicData, *meta_data*: rti.connexdds.FilterSampleInfo) → rti.connexdds.CookieVector

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

writer_finalize (*self*: rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object], *cookie*: rti.connexdds.Cookie) → None

A writer-side filtering API to clean up a previously compiled instance of the content filter.

writer_return_loan (*self*: rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object], *cookies*: rti.connexdds.CookieVector) → None

A writer-side filtering API to return the loan on the list of DataReaders returned by `writer_evaluate`.

class WriterContentFilterHelper

Bases: *WriterContentFilter*

__init__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper) → None

__module__ = 'rti.connexdds'

add_cookie (*self*: rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper, *cookie*: rti.connexdds.Cookie) → None

A helper function which will add a Cookie to the Cookie sequence that is then returned by the `writer_evaluate` function.

writer_evaluate_helper (*self*: rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper, *writer_filter_data*: *Optional[object]*, *sample*: rti.connexdds.SubscriptionBuiltinTopicData, *meta_data*: rti.connexdds.FilterSampleInfo) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

__eq__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData) → None

Create a default SubscriptionBuiltinTopicData.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.SubscriptionBuiltinTopicData, *arg0*: rti.connexdds.SubscriptionBuiltinTopicData) → bool

Test for inequality.

builtin_topic_name = 'DCPSSubscription'

property content_filter_property

The custom multicast locators that the endpoint can specify.

property data_tag

The DataTag policy of the corresponding DataReader.

property deadline

The Deadline policy of the corresponding DataReader.

property destination_order

The DestinationOrder policy of the corresponding DataReader.

property disable_positive_acks

See whether or not a matching DataReader will send positive acknowledgments for reliability.

property durability

The Durability policy of the corresponding DataReader.

property group_data

The GroupData policy of the corresponding DataReader's Subscriber.

property key

The DCPS key to distinguish entries.

property latency_budget

The LatencyBudget policy of the corresponding DataReader.

property liveliness

The Liveliness policy of the corresponding DataReader.

property multicast_locators

The custom multicast locators that the endpoint can specify.

property ownership

The Ownership policy of the corresponding DataReader.

property participant_key

The DCPS key of the DomainParticipant to which the DataReader belongs.

property partition

The Partition policy of the corresponding DataReader's Subscriber.

property presentation

The Presentation policy of the corresponding DataReader's Subscriber.

property product_version

The current version for RTI Connex.

property property

The propagated name-value properties of the corresponding DataReader.

property reliability

The Reliability policy of the corresponding DataReader.

property representation

The Representation policy of the corresponding DataReader.

property rtps_protocol_version

The version number of the RTPS wire protocol used.

property rtps_vendor_id

The ID of the vendor implementing the RTPS wire protocol.

property service

The Service policy

property subscriber_key

The DCPS key of the Publisher to which the DataReader belongs.

property subscription_name

The subscription name and role name.

property time_based_filter

The TimeBasedFilter policy of the corresponding DataReader.

property topic_data

The TopicData policy of the corresponding DataReader's Topic.

property topic_name

The name of the related Topic.

property type

The type.

property type_name

The name of the type attached to the Topic.

property unicast_locators

The custom unicast locators that the endpoint can specify.

property user_data

The UserData policy of the corresponding DataReader.

property virtual_guid

The virtual Guid associated to the DataReader.

class `rti.connextdds.SubscriptionBuiltinTopicDataSeq`

Bases: `pybind11_object`

__add__ (*self*: `rti.connextdds.SubscriptionBuiltinTopicDataSeq`, *arg0*: `rti.connextdds.SubscriptionBuiltinTopicDataSeq`) → `rti.connextdds.SubscriptionBuiltinTopicDataSeq`

__bool__ (*self*: `rti.connextdds.SubscriptionBuiltinTopicDataSeq`) → `bool`
 Check whether the list is nonempty

__contains__ (*self*: `rti.connextdds.SubscriptionBuiltinTopicDataSeq`, *x*: `rti.connextdds.SubscriptionBuiltinTopicData`) → `bool`

Return true the container contains *x*

`__delitem__` (*args, **kwargs)

Overloaded function.

1. `__delitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: int) -> None`

Delete the list elements at index `i`

2. `__delitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicDataSeq) → bool

`__getitem__` (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, s: slice) -> rti.connextdds.SubscriptionBuiltinTopicDataSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: int) -> rti.connextdds.SubscriptionBuiltinTopicData`

`__hash__` = None

`__iadd__` (self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicDataSeq) → rti.connextdds.SubscriptionBuiltinTopicDataSeq

`__imul__` (self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: int) → rti.connextdds.SubscriptionBuiltinTopicDataSeq

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq) -> None`
2. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: rti.connextdds.SubscriptionBuiltinTopicDataSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: Iterable) -> None`

`__iter__` (self: rti.connextdds.SubscriptionBuiltinTopicDataSeq) → Iterator

`__len__` (self: rti.connextdds.SubscriptionBuiltinTopicDataSeq) → int

`__module__` = 'rti.connextdds'

`__mul__` (self: rti.connextdds.SubscriptionBuiltinTopicDataSeq, arg0: int) → rti.connextdds.SubscriptionBuiltinTopicDataSeq

__ne__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *arg0*: rti.connexdds.SubscriptionBuiltinTopicDataSeq) → bool

__rmul__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *arg0*: int) → rti.connexdds.SubscriptionBuiltinTopicDataSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *arg0*: int, *arg1*: rti.connexdds.SubscriptionBuiltinTopicData) -> None
2. **__setitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *arg0*: slice, *arg1*: rti.connexdds.SubscriptionBuiltinTopicDataSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *x*: rti.connexdds.SubscriptionBuiltinTopicData) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq) → None

Clear the contents

count (*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *x*: rti.connexdds.SubscriptionBuiltinTopicData) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *L*: rti.connexdds.SubscriptionBuiltinTopicDataSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *i*: int, *x*: rti.connexdds.SubscriptionBuiltinTopicData) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq) -> rti.connexdds.SubscriptionBuiltinTopicData

Remove and return the last item

2. **pop**(*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *i*: int) -> rti.connexdds.SubscriptionBuiltinTopicData

Remove and return the item at index *i*

remove (*self*: rti.connexdds.SubscriptionBuiltinTopicDataSeq, *x*:
rti.connexdds.SubscriptionBuiltinTopicData) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0*:
rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) →
rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq

__bool__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *x*:
Tuple[rti.connexdds.SubscriptionBuiltinTopicData, rti.connexdds.Time]) →
bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0*: int)
-> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0*:
slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0*:
rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *s*: slice) ->
rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0*: int)
-> *Tuple*[rti.connexdds.SubscriptionBuiltinTopicData, rti.connexdds.Time]

__hash__ = None

__iadd__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0*:
rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) →
rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq

__imul__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0*: int) → *rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq*

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) -> None
2. **__init__**(self: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) -> None

Copy constructor

3. **__init__**(self: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: Iterable) -> None

__iter__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) → Iterator

__len__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0*: int) → *rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq*

__ne__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__rmul__ (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *arg0*: int) → *rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq*

__setitem__ (*args, **kwargs)

Overloaded function.

1. **__setitem__**(self: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: int, arg1: Tuple[rti.connexdds.SubscriptionBuiltinTopicData, rti.connexdds.Time]) -> None
2. **__setitem__**(self: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, arg0: slice, arg1: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq, *x*: Tuple[rti.connexdds.SubscriptionBuiltinTopicData, rti.connexdds.Time]) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq) → None

Clear the contents

count (*self*: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *x*:
Tuple[rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.Time]) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *L*: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) -> None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *i*: int, *x*:
Tuple[rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.Time]) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq) -> *Tuple*[rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.Time]

Remove and return the last item

2. pop(*self*: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *i*: int) -> *Tuple*[rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.Time]

Remove and return the item at index *i*

remove (*self*: rti.connextdds.SubscriptionBuiltinTopicDataTimestampedSeq, *x*:
Tuple[rti.connextdds.SubscriptionBuiltinTopicData, rti.connextdds.Time]) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connextdds.**SubscriptionMatchedStatus**

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connextdds'

property **current_count**

The number of DataWriters that are currently matched with this DataReader.

property **current_count_change**

The delta for the current count since the last time the listener fired or the status was read.

property **current_count_peak**

The highest value that the current count has reached.

property last_publication_handle

A handle to the DataWriter that caused the most recent change to to this status.

property total_count

Total count of times the DataReader matched with a DataWriter.

property total_count_change

The delta in the total count since the last time the listener fired or the status was read.

class rti.connexdds.SuspendedPublication

Bases: pybind11_object

__enter__ (*self*: rti.connexdds.SuspendedPublication) → *rti.connexdds.SuspendedPublication*

Context manage the SuspendedPublication.

__exit__ (*self*: rti.connexdds.SuspendedPublication, *arg0*: object, *arg1*: object, *arg2*: object) → None

__init__ (*self*: rti.connexdds.SuspendedPublication, *publisher*: rti.connexdds.Publisher) → None

Suspends the publications of the publisher.

__module__ = 'rti.connexdds'

resume (*self*: rti.connexdds.SuspendedPublication) → None

Indicates that the application has completed these changes.

class rti.connexdds.SyslogVerbosity

Bases: pybind11_object

Members:

SILENT : Silent verbosity level

EMERGENCY : Emergency verbosity level

ALERT : Alert verbosity level

CRITICAL : Critical verbosity level

ERROR : Error verbosity level

WARNING : Warning verbosity level

NOTICE : Notice verbosity level

INFORMATIONAL : Informational verbosity level

DEBUG : Debug verbosity level

ALERT = <SyslogVerbosity.ALERT: 3>

CRITICAL = <SyslogVerbosity.CRITICAL: 7>

DEBUG = <SyslogVerbosity.DEBUG: 255>

```

EMERGENCY = <SyslogVerbosity.EMERGENCY: 1>
ERROR = <SyslogVerbosity.ERROR: 15>
INFORMATIONAL = <SyslogVerbosity.INFORMATIONAL: 127>
NOTICE = <SyslogVerbosity.NOTICE: 63>
SILENT = <SyslogVerbosity.SILENT: 0>
WARNING = <SyslogVerbosity.WARNING: 31>

__eq__ (self: object, other: object) → bool

__getstate__ (self: object) → int

__hash__ (self: object) → int

__index__ (self: rti.connexdds.SyslogVerbosity) → int

__init__ (self: rti.connexdds.SyslogVerbosity, value: int) → None

__int__ (self: rti.connexdds.SyslogVerbosity) → int

__members__ = {'ALERT': <SyslogVerbosity.ALERT: 3>, 'CRITICAL':
<SyslogVerbosity.CRITICAL: 7>, 'DEBUG': <SyslogVerbosity.DEBUG:
255>, 'EMERGENCY': <SyslogVerbosity.EMERGENCY: 1>, 'ERROR':
<SyslogVerbosity.ERROR: 15>, 'INFORMATIONAL':
<SyslogVerbosity.INFORMATIONAL: 127>, 'NOTICE':
<SyslogVerbosity.NOTICE: 63>, 'SILENT': <SyslogVerbosity.SILENT:
0>, 'WARNING': <SyslogVerbosity.WARNING: 31>}

__module__ = 'rti.connexdds'

__ne__ (self: object, other: object) → bool

__repr__ (self: object) → str

__setstate__ (self: rti.connexdds.SyslogVerbosity, state: int) → None

__str__ ()
    name(self: handle) -> str

property name
property value

```

```
class rti.connexdds.SystemResourceLimits
```

```
Bases: pybind11_object
```

```
__eq__ (self: rti.connexdds.SystemResourceLimits, arg0: rti.connexdds.SystemResourceLimits)
    → bool
```

```
    Test for equality.
```


__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connextdds.SystemResourceLimits) -> None

Creates a SystemResourceLimits qos policy with default values.

2. **__init__**(self: rti.connextdds.SystemResourceLimits, max_objects_per_thread: int) -> None

Creates a SystemResourceLimits qos policy with the provided max_objects_per_thread.

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.SystemResourceLimits, arg0: rti.connextdds.SystemResourceLimits)
→ bool

Test for inequality.

property initial_objects_per_thread

The number of objects per thread for a DomainParticipantFactory for which infrastructure will initially be allocated.

property max_objects_per_thread

The maximum number of objects that can be stored per thread.

class rti.connextdds.ThreadContext

Bases: pybind11_object

__enter__ (self: rti.connextdds.ThreadContext) → rti.connextdds.ThreadContext

__exit__ (self: rti.connextdds.ThreadContext, arg0: object, arg1: object, arg2: object) → None

__init__ (self: rti.connextdds.ThreadContext) → None

Creates a thread context that will unregister the thread on exit.

__module__ = 'rti.connextdds'

class rti.connextdds.ThreadSettings

Bases: pybind11_object

__eq__ (self: rti.connextdds.ThreadSettings, arg0: rti.connextdds.ThreadSettings) → bool

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connextdds.ThreadSettings) -> None

Create a ThreadSettings object with default settings.

2. **__init__**(self: rti.connextdds.ThreadSettings, mask: rti.connextdds.ThreadSettingsKind-Mask, priority: int, stack_size: int, cpu_list: rti.connextdds.Int32Vector, cpu_rotation: rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind) -> None

Create a ThreadSettings object with the specified parameters.

```
__module__ = 'rti.connextdds'
```

```
__ne__ (self: rti.connextdds.ThreadSettings, arg0: rti.connextdds.ThreadSettings) → bool
```

```
property cpu_list
```

Get/set a copy of the list of CPUs available to the thread.

```
property cpu_rotation
```

Get/set a copy of the thread settings mask.

```
property mask
```

Get/set a copy of the thread settings mask.

```
property priority
```

Get/set a copy of the thread priority.

```
property stack_size
```

Get/set a copy of the thread stack size.

```
class rti.connextdds.ThreadSettingsCpuRotationKind
```

Bases: pybind11_object

```
NO_ROTATION = <ThreadSettingsCpuRotationKind.NO_ROTATION: 0>
```

```
ROUND_ROBIN = <ThreadSettingsCpuRotationKind.ROUND_ROBIN: 1>
```

```
class ThreadSettingsCpuRotationKind
```

Bases: pybind11_object

Members:

NO_ROTATION : Any thread controlled by this QoS can run on any listed processor, as determined by OS scheduling.

ROUND_ROBIN : Threads controlled by this QoS will be assigned one processor from the list in round-robin order.

```
NO_ROTATION = <ThreadSettingsCpuRotationKind.NO_ROTATION: 0>
```

```
ROUND_ROBIN = <ThreadSettingsCpuRotationKind.ROUND_ROBIN: 1>
```

```
__eq__ (self: object, other: object) → bool
```

```
__getstate__ (self: object) → int
```

```
__hash__ (self: object) → int
```

```
__index__ (self: rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind) → int
```

```

__init__(self:
    rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind,
    value: int) → None

__int__(self:
    rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind)
    → int

__members__ = {'NO_ROTATION':
<ThreadSettingsCpuRotationKind.NO_ROTATION: 0>, 'ROUND_ROBIN':
<ThreadSettingsCpuRotationKind.ROUND_ROBIN: 1>}

__module__ = 'rti.connexdds'

__ne__(self: object, other: object) → bool

__repr__(self: object) → str

__setstate__(self: rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRo-
    tationKind, state: int) →
    None

__str__()
    name(self: handle) -> str

property name

property value

__eq__(self: rti.connexdds.ThreadSettingsCpuRotationKind, arg0:
    rti.connexdds.ThreadSettingsCpuRotationKind) → bool

    Apply operator to underlying enumerated values.

__ge__(self: rti.connexdds.ThreadSettingsCpuRotationKind, arg0:
    rti.connexdds.ThreadSettingsCpuRotationKind) → bool

    Apply operator to underlying enumerated values.

__gt__(self: rti.connexdds.ThreadSettingsCpuRotationKind, arg0:
    rti.connexdds.ThreadSettingsCpuRotationKind) → bool

    Apply operator to underlying enumerated values.

__hash__ = None

__init__(*args, **kwargs)
    Overloaded function.

    1. __init__(self: rti.connexdds.ThreadSettingsCpuRotationKind) -> None
    Initializes enum to 0.

    2. __init__(self: rti.connexdds.ThreadSettingsCpuRotationKind, arg0: rti.connexdds.Thread-
        SettingsCpuRotationKind.ThreadSettingsCpuRotationKind) -> None

```

Copy constructor.

```
__int__ (self: rti.connextdds.ThreadSettingsCpuRotationKind) →
    rti.connextdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind
```

Int conversion.

```
__le__ (self: rti.connextdds.ThreadSettingsCpuRotationKind, arg0:
    rti.connextdds.ThreadSettingsCpuRotationKind) → bool
```

Apply operator to underlying enumerated values.

```
__lt__ (self: rti.connextdds.ThreadSettingsCpuRotationKind, arg0:
    rti.connextdds.ThreadSettingsCpuRotationKind) → bool
```

Apply operator to underlying enumerated values.

```
__module__ = 'rti.connextdds'
```

```
__ne__ (self: rti.connextdds.ThreadSettingsCpuRotationKind, arg0:
    rti.connextdds.ThreadSettingsCpuRotationKind) → bool
```

Apply operator to underlying enumerated values.

```
__str__ (self: rti.connextdds.ThreadSettingsCpuRotationKind) → str
    String conversion.
```

property underlying

Retrieves the actual enumerated value.

```
class rti.connextdds.ThreadSettingsKindMask
```

Bases: pybind11_object

```
CANCEL_ASYNCHRONOUS = 0000
```

```
FLOATING_POINT = 0001
```

```
PRIORITY_ENFORCE = 0000
```

```
REALTIME_PRIORITY = 1000
```

```
STDIO = 0010
```

```
__and__ (self: rti.connextdds.ThreadSettingsKindMask, arg0:
    rti.connextdds.ThreadSettingsKindMask) → rti.connextdds.ThreadSettingsKindMask
```

Bitwise logical AND of masks.

```
__bool__ (self: rti.connextdds.ThreadSettingsKindMask) → bool
```

Test if any bits are set.

```
__contains__ (self: rti.connextdds.ThreadSettingsKindMask, arg0:
    rti.connextdds.ThreadSettingsKindMask) → bool
```

```
__eq__ (self: rti.connextdds.ThreadSettingsKindMask, arg0:
    rti.connextdds.ThreadSettingsKindMask) → bool
```

Compare masks for equality.

__getitem__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: int) → bool
Get individual mask bit.

__hash__ = None

__iand__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: rti.connexdds.ThreadSettingsKindMask) → rti.connexdds.ThreadSettingsKindMask
Set mask to logical AND with another mask.

__ilshift__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: int) → rti.connexdds.ThreadSettingsKindMask
Left shift bits in mask.

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: rti.connexdds.ThreadSettingsKindMask) -> None
Create a ThreadSettingsKindMask with default thread settings.
2. **__init__**(*self*: rti.connexdds.ThreadSettingsKindMask, *value*: int) -> None
Creates a mask from the bits in an integer.

__int__ (*self*: rti.connexdds.ThreadSettingsKindMask) → int
Convert mask to int.

__ior__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: rti.connexdds.ThreadSettingsKindMask) → rti.connexdds.ThreadSettingsKindMask
Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: int) → rti.connexdds.ThreadSettingsKindMask
Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: rti.connexdds.ThreadSettingsKindMask) → rti.connexdds.ThreadSettingsKindMask
Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: int) → rti.connexdds.ThreadSettingsKindMask
Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: rti.connexdds.ThreadSettingsKindMask) → bool
Compare masks for inequality.

__or__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: rti.connexdds.ThreadSettingsKindMask) → rti.connexdds.ThreadSettingsKindMask
Bitwise logical OR of masks.

__repr__ (*self*: rti.connexdds.ThreadSettingsKindMask) → str

Convert mask to string.

__rshift__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: int) →
rti.connexdds.ThreadSettingsKindMask

Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*: int, *arg1*: bool) → None

Set individual mask bit

__str__ (*self*: rti.connexdds.ThreadSettingsKindMask) → str

Convert mask to string.

__xor__ (*self*: rti.connexdds.ThreadSettingsKindMask, *arg0*:
rti.connexdds.ThreadSettingsKindMask) → *rti.connexdds.ThreadSettingsKindMask*

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (**args*, ***kwargs*)

Overloaded function.

1. flip(*self*: rti.connexdds.ThreadSettingsKindMask) -> rti.connexdds.ThreadSettingsKindMask

Flip all bits in the mask.

2. flip(*self*: rti.connexdds.ThreadSettingsKindMask, *pos*: int) -> rti.connexdds.ThreadSettingsKindMask

Flip the mask bit at the specified position.

reset (**args*, ***kwargs*)

Overloaded function.

1. reset(*self*: rti.connexdds.ThreadSettingsKindMask) -> rti.connexdds.ThreadSettingsKindMask

Clear all bits in the mask.

2. reset(*self*: rti.connexdds.ThreadSettingsKindMask, *pos*: int) -> rti.connexdds.ThreadSettingsKindMask

Clear the mask bit at the specified position.

set (**args*, ***kwargs*)

Overloaded function.

1. set(*self*: rti.connexdds.ThreadSettingsKindMask) -> rti.connexdds.ThreadSettingsKindMask

Set all bits in the mask.

2. `set(self: rti.connextdds.ThreadSettingsKindMask, pos: int, value: bool = True) -> rti.connextdds.ThreadSettingsKindMask`

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (*self*: rti.connextdds.ThreadSettingsKindMask, *pos*: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (*self*: rti.connextdds.ThreadSettingsKindMask) → bool

Test if all bits are set.

test_any (*self*: rti.connextdds.ThreadSettingsKindMask) → bool

Test if any bits are set.

test_none (*self*: rti.connextdds.ThreadSettingsKindMask) → bool

Test if none of the bits are set.

class rti.connextdds.Time

Bases: pybind11_object

__add__ (*self*: rti.connextdds.Time, *arg0*: rti.connextdds.Duration) → rti.connextdds.Time

Add a Time and Duration together.

__eq__ (*self*: rti.connextdds.Time, *arg0*: rti.connextdds.Time) → bool

Check if this Time is equal to another.

__ge__ (*self*: rti.connextdds.Time, *arg0*: rti.connextdds.Time) → bool

Check if this Time is greater than or equal another.

__gt__ (*self*: rti.connextdds.Time, *arg0*: rti.connextdds.Time) → bool

Check if this Time is greater than another.

__hash__ = None

__iadd__ (*self*: rti.connextdds.Time, *arg0*: rti.connextdds.Duration) → rti.connextdds.Time

Add a Duration to this Time object.

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.Time) -> None`

Create a default Time object. The constructed Time object will represent 0 seconds and 0 nanoseconds.

2. `__init__(self: rti.connextdds.Time, sec: int, nanosec: int = 0) -> None`

Create a Time object. The constructed Time object will represent the given amount of time.

3. `__init__(self: rti.connextdds.Time, other: rti.connextdds.Time) -> None`

Create a copy of a Time.

4. `__init__(self: rti.connextdds.Time, arg0: datetime.datetime) -> None`

Create a Time object from a timestamp.

`__isub__(self: rti.connextdds.Time, arg0: rti.connextdds.Duration) → rti.connextdds.Time`

Subtract a Duration from this Time object.

`__le__(self: rti.connextdds.Time, arg0: rti.connextdds.Time) → bool`

Check if this Time is less than or equal another.

`__lt__(self: rti.connextdds.Time, arg0: rti.connextdds.Time) → bool`

Check if this Time is less than another.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.Time, arg0: rti.connextdds.Time) → bool`

Check if this Time is not equal to another.

`__radd__(self: rti.connextdds.Time, arg0: rti.connextdds.Duration) → rti.connextdds.Time`

Add a Time and Duration together.

`__sub__(*args, **kwargs)`

Overloaded function.

1. `__sub__(self: rti.connextdds.Time, arg0: rti.connextdds.Duration) -> rti.connextdds.Time`

Subtract a Duration from a Time.

2. `__sub__(self: rti.connextdds.Time, arg0: rti.connextdds.Time) -> rti.connextdds.Duration`

Calculate the duration between two times.

`compare(self: rti.connextdds.Time, other: rti.connextdds.Time) → int`

Compare two Time objects.

`static from_microseconds (microseconds: int) → rti.connextdds.Time`

Create a Time object from microseconds.

`static from_milliseconds (milliseconds: int) → rti.connextdds.Time`

Create a Time object from millisecs.

`static from_seconds (seconds: float) → rti.connextdds.Time`

Create a Time object from seconds.

`invalid = <rti.connextdds.Time object>`

`maximum = <rti.connextdds.Time object>`

property nanosec

The number of nanoseconds that are represented by this Time object.

property sec

The number of seconds that are represented by this Time object.

to_microseconds (*self*: rti.connextdds.Time) → int

Convert this Time to microseconds.

to_milliseconds (*self*: rti.connextdds.Time) → int

Convert this Time to milliseconds.

to_seconds (*self*: rti.connextdds.Time) → float

Convert this Time to seconds.

zero = <rti.connextdds.Time object>

class rti.connextdds.TimeBasedFilter

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.TimeBasedFilter, *arg0*: rti.connextdds.TimeBasedFilter) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.TimeBasedFilter) -> None

Creates the default time-based filter.

2. **__init__**(*self*: rti.connextdds.TimeBasedFilter, *min_separation*: rti.connextdds.Duration) -> None

Creates a policy with the specified minimum separation.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.TimeBasedFilter, *arg0*: rti.connextdds.TimeBasedFilter) → bool

Test for inequality.

property **minimum_separation**

The minimum separation between subsequent samples.

exception rti.connextdds.TimeoutError

Bases: *Exception*

__module__ = 'rti.connextdds'

class rti.connextdds.Topic

Bases: *ITopicDescription*, *IAnyTopic*

__eq__ (*self*: rti.connextdds.Topic, *arg0*: rti.connextdds.Topic) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.Topic, entity: rti.connexdds.IEntity) -> None

Cast an Entity to a Topic.

2. **__init__**(self: rti.connexdds.Topic, topic_description: rti.connexdds.ITopicDescription) -> None

Cast an ITopicDescription to a Topic.

3. **__init__**(self: rti.connexdds.Topic, topic: rti.connexdds.IAnyTopic) -> None

Create a typed Topic from an AnyTopic.

4. **__init__**(self: rti.connexdds.Topic, participant: rti.connexdds.DomainParticipant, topic_name: str, type: object, qos: rti.connexdds.TopicQos = None, listener: rti.connexdds.TopicListener = None, mask: rti.connexdds.StatusMask = StatusMask.ALL, type_name: str = None) -> None

Create a Topic for an @idl.struct or @idl.union type.

The participant, topic_name and type arguments are required. The type_name, qos, listener, mask arguments are optional. The type_name argument can be used to register the type with a name different from the class name.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.Topic, arg0: rti.connexdds.Topic) → bool

Test for inequality.

static find (participant: rti.connexdds.DomainParticipant, name: str) → Optional[rti.connexdds.Topic]

Look up a Topic by its name in the DomainParticipant.

property inconsistent_topic_status

Get a copy of the current InconsistentTopicStatus for this Topic.

property listener

The listener.

property qos

Get the TopicQos for this Topic.

This property's getter returns a deep copy.

set_listener (self: rti.connexdds.Topic, listener: rti.connexdds.TopicListener, event_mask: rti.connexdds.StatusMask) → None

Set the listener and event mask.

property type

Get the type associated with the topic.

property type_support

Get the type_support object associated with the topic.

class rti.connexdds.TopicBuiltinTopicData

Bases: pybind11_object

class ContentFilter

Bases: ContentFilterBase

__init__ (self: rti.connexdds.TopicBuiltinTopicData.ContentFilter) → None

__module__ = 'rti.connexdds'

compile (self: rti.connexdds.TopicBuiltinTopicData.ContentFilter, expression: str, parameters: rti.connexdds.StringSeq, type_code: Optional[rti.connexdds.DynamicType], type_class_name: str, old_compile_data: Optional[object]) → Optional[object]

Compile an instance of the content filter according to the filter expression and parameters of the given data type.

evaluate (self: rti.connexdds.TopicBuiltinTopicData.ContentFilter, compile_data: Optional[object], sample: rti.connexdds.TopicBuiltinTopicData, meta_data: rti.connexdds.FilterSampleInfo) → bool

Evaluate whether the sample is passing the filter or not according to the sample content.

finalize (self: rti.connexdds.TopicBuiltinTopicData.ContentFilter, compile_data: Optional[object]) → None

A previously compiled instance of the content filter is no longer in use and resources can now be cleaned up.

class ContentFilteredTopic

Bases: ITopicDescription, IAnyTopic

__eq__ (self: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic, arg0: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic, topic: rti.connexdds.TopicBuiltinTopicData.Topic, name: str, contentfilter: rti.connexdds.Filter) -> None

Create a ContentFilteredTopic with a name and Filter.

2. **__init__**(self: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic, topic_description: rti.connexdds.TopicBuiltinTopicData.ITopicDescription) -> None

Cast a TopicDescription to a ContentFilteredTopic.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic, *arg0*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic) → bool

Test for inequality.

append_to_expression_parameter (*self*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic, *index*: int, *extension*: str) → None

Append the extension to the end of parameter at the provided index, separated by a comma.

property filter_expression

Get the filter expression

property filter_parameters

Get/set the filter parameters.

static find (*participant*: rti.connexdds.DomainParticipant, *name*: str) → Optional[rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic]

Look up a ContentFilteredTopic by its name in the DomainParticipant.

remove_from_expression_parameter (*self*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic, *index*: int, *remove_term*: str) → None

Removes the specified term from the parameter at the provided index.

set_filter (*self*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic, *arg0*: rti.connexdds.Filter) → None

Set the filter.

property topic

Get the underlying Topic.

class ContentFilteredTopicSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) → rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq

__bool__ (*self*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, *arg0*: int) → None

Delete the list elements at index *i*

2. `__delitem__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) -> bool`

`__getitem__(*args, **kwargs)`

Overloaded function.

1. `__getitem__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, s: slice) -> rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic`

`__hash__ = None`

`__iadd__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) -> rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq`

`__imul__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) -> None`

2. `__init__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) -> Iterator`

`__len__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) -> int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq`

`__ne__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) -> bool`

`__rmul__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq`

`__setitem__(*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: int, arg1: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic) -> None`
 2. `__setitem__(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, arg0: slice, arg1: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) -> None`
- Assign list elements using a slice object

append (*self*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic) → None

Add an item to the end of the list

clear (*self*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) → None

Clear the contents

count (*self*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. `extend(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, L: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, *i*: int, *x*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq) -> rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic`

Remove and return the last item

2. `pop(self: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, i: int) -> rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq, *x*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class DataReader

Bases: *IDataReader*

class Selector

Bases: *pybind11_object*

__init__ (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector, *datareader*: rti.connexdds.TopicBuiltinTopicData.DataReader) → None

Create a Selector for a DataReader to read/take based on selected conditions

__module__ = 'rti.connexdds'

condition (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector, *condition*: rti.connexdds.IReadCondition) → *rti.connexdds.TopicBuiltinTopicData.DataReader.Selector*

Select samples based on a ReadCondition.

content (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector, *query*: rti.connexdds.Query) → *rti.connexdds.TopicBuiltinTopicData.DataReader.Selector*

Select samples based on a Query.

instance (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector, *handle*: rti.connexdds.InstanceHandle) → *rti.connexdds.TopicBuiltinTopicData.DataReader.Selector*

Select a specific instance to read/take.

max_samples (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector, *max*: int) → *rti.connexdds.TopicBuiltinTopicData.DataReader.Selector*

Limit the number of samples read/taken by the Select.

next_instance (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector, *previous*: rti.connexdds.InstanceHandle) → *rti.connexdds.TopicBuiltinTopicData.DataReader.Selector*

Select the instance after the specified instance to read/take.

read (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector) → list

Read copies of available samples (data and info) based on the Selector settings.

read_data (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector) → list

Read copies of available valid data based on the Selector settings.

read_loaned (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector) → *rti.connexdds.TopicBuiltinTopicData.LoanedSamples*

Take available samples (data and info) based on the Selector settings and return them in a loaned container.

state (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector, *state*: rti.connexdds.DataState) → *rti.connexdds.TopicBuiltinTopicData.DataReader.Selector*

Select samples with a specified data state.

take (*self*: rti.connexdds.TopicBuiltinTopicData.DataReader.Selector) → list

Take copies of available samples (data and info) based on the Selector settings.

take_data (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader.Selector) → list
 Take copies of available valid data based on the Selector settings.

take_loaned (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader.Selector) →
rti.connextdds.TopicBuiltinTopicData.LoanedSamples

Read available samples (data and info) based on the Selector settings and return them in a loaned container.

__enter__ (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader) →
rti.connextdds.TopicBuiltinTopicData.DataReader

Enter a context for this DataReader, to be cleaned up on exiting context

__eq__ (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *arg0*:
 rti.connextdds.TopicBuiltinTopicData.DataReader) → bool

Test for equality.

__exit__ (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *arg0*: object, *arg1*: object,
arg2: object) → None

Exit the context for this DataReader, cleaning up resources.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *topic*: rti.connextdds.TopicBuiltinTopicData.Topic) -> None

Create a DataReader in the implicit subscriber with default QoS.

2. **__init__**(*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *topic*: rti.connextdds.TopicBuiltinTopicData.Topic, *qos*: rti.connextdds.DataReaderQos, *listener*: rti.connextdds.TopicBuiltinTopicData.DataReaderListener = None, *mask*: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader in the implicit subscriber with specific QoS and a listener.

3. **__init__**(*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *cft*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic) -> None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with default QoS.

4. **__init__**(*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *cft*: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic, *qos*: rti.connextdds.DataReaderQos, *listener*: rti.connextdds.TopicBuiltinTopicData.DataReaderListener = None, *mask*: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader with a ContentFilteredTopic in the implicit subscriber with specific QoS.

5. **__init__**(*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *subscriber*: rti.connextdds.Subscriber, *topic*: rti.connextdds.TopicBuiltinTopicData.Topic) -> None

Create a DataReader.

6. **__init__**(*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *subscriber*: rti.connextdds.Subscriber, *topic*: rti.connextdds.TopicBuiltinTopicData.Topic, *qos*: rti.connextdds.DataReaderQos, *listener*: rti.connextdds.TopicBuiltinTopicData.DataReaderListener = None, *mask*: rti.connextdds.StatusMask = StatusMask.ALL) -> None

Create a DataReader in a subscriber with specific QoS and a listener.

7. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataReader, subscriber: rti.connextdds.Subscriber, cft: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic) -> None`

Create a DataReader with a ContentFilteredTopic in a subscriber with default QoS.

8. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataReader, subscriber: rti.connextdds.Subscriber, cft: rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopic, qos: rti.connextdds.DataReaderQos, listener: rti.connextdds.TopicBuiltinTopicData.DataReaderListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Create a DataReader with a ContentFilteredTopic in a subscriber with specific QoS.

9. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataReader, reader: rti.connextdds.IAnyDataReader) -> None`

Get a typed DataReader from an AnyDataReader.

10. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataReader, entity: rti.connextdds.IEntity) -> None`

Get a typed DataReader from an Entity.

`__lshift__ (self: rti.connextdds.TopicBuiltinTopicData.DataReader, arg0: rti.connextdds.DataReaderQos) -> rti.connextdds.TopicBuiltinTopicData.DataReader`

Set the DataReaderQos for this DataReader.

`__module__ = 'rti.connextdds'`

`__ne__ (self: rti.connextdds.TopicBuiltinTopicData.DataReader, arg0: rti.connextdds.TopicBuiltinTopicData.DataReader) -> bool`

Test for inequality.

`__rshift__ (self: rti.connextdds.TopicBuiltinTopicData.DataReader, arg0: rti.connextdds.DataReaderQos) -> rti.connextdds.TopicBuiltinTopicData.DataReader`

Get the DataReaderQos from this DataReader

`acknowledge_all (*args, **kwargs)`

Overloaded function.

1. `acknowledge_all(self: rti.connextdds.TopicBuiltinTopicData.DataReader) -> None`

Acknowledge all previously accessed samples.

2. `acknowledge_all(self: rti.connextdds.TopicBuiltinTopicData.DataReader, arg0: rti.connextdds.AckResponseData) -> None`

Acknowledge all previously accessed samples.

`acknowledge_sample (*args, **kwargs)`

Overloaded function.

1. `acknowledge_sample(self: rti.connextdds.TopicBuiltinTopicData.DataReader, sample_info: rti.connextdds.SampleInfo) -> None`

Acknowledge a single sample.

2. `acknowledge_sample(self: rti.connextdds.TopicBuiltinTopicData.DataReader, sample_info: rti.connextdds.SampleInfo, ack_response_data: rti.connextdds.AckResponseData) -> None`

Acknowledge a single sample with ack response data.

close (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader) → None

Close this DataReader.

property datareader_cache_status

Get the DataReaderCacheStatus for the DataReader.

property datareader_protocol_status

Get the DataReaderProtocolStatus for the DataReader.

property default_filter_state

Returns the filter state for the read/take operations.

static find_all_by_topic (*subscriber*: rti.connextdds.Subscriber, *topic_name*: str) → *rti.connextdds.TopicBuiltinTopicData.DataReaderSeq*

Retrieve all DataReaders for the given topic name in the subscriber.

static find_by_name (*args, **kwargs)

Overloaded function.

1. find_by_name(participant: rti.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.TopicBuiltinTopicData.DataReader]

Find DataReader in DomainParticipant with the DataReader's name, returning the first found.

2. find_by_name(subscriber: rti.connextdds.Subscriber, name: str) -> Optional[rti.connextdds.TopicBuiltinTopicData.DataReader]

Find DataReader in Subscriber with the DataReader's name, returning the first found.

static find_by_topic (*subscriber*: rti.connextdds.Subscriber, *name*: str) → Optional[*rti.connextdds.TopicBuiltinTopicData.DataReader*]

Find DataReader in Subscriber with a topic name, returning the first found.

is_matched_publication_alive (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *arg0*: rti.connextdds.InstanceHandle) → bool

Check if a matched publication is alive.

key_value (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.TopicBuiltinTopicData*

Retrieve the instance key that corresponds to an instance handle.

property listener

Gets or sets the listener with StatusMask.ALL

property liveliness_changed_status

Get the LivelinessChangedStatus for this DataReader.

lookup_instance (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *key_holder*: rti.connextdds.TopicBuiltinTopicData) → *rti.connextdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

matched_publication_data (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.PublicationBuiltinTopicData*

Get the PublicationBuiltinTopicData for a publication matched to this DataReader.

matched_publication_datareader_protocol_status (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *publication_handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.DataReaderProtocolStatus*

Get the DataReaderProtocolStatus for the DataReader based on the matched publication handle.

matched_publication_participant_data (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a publication matched to this DataReader.

property matched_publications

Get a copy of the list of the currently matched publication handles.

property qos

The DataReaderQos for this DataReader.

This property's getter returns a deep copy.

read (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader) → list

Read copies of all available samples (data and info).

read_data (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader) → list

Read copies of all available valid data.

read_loaned (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader) →

rti.connextdds.TopicBuiltinTopicData.LoanedSamples

Read all available samples (data and info) and return them in a loaned container.

property requested_deadline_missed_status

Get the RequestedDeadlineMissed status for the DataReader

property requested_incompatible_qos_status

Get the RequestedIncompatibleQosStatus for the DataReader.

property sample_lost_status

Get the SampleLostStatus for the DataReader.

property sample_rejected_status

Get the SampleRejectedStatus for the DataReader.

select (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader) →
 dds::sub::DataReader<dds::topic::TTopicBuiltinTopicData<rti::topic::TopicBuiltin-
 TopicDataImpl>,
 rti::sub::DataReaderImpl>::Selector

Get a Selector to perform complex data selections, such as per-instance selection, content, and status filtering.

set_listener (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader, *listener*:
 rti.connextdds.TopicBuiltinTopicData.DataReaderListener, *event_mask*:
 rti.connextdds.StatusMask) → None

Set the listener and associated event mask.

property subscriber

Returns the parent Subscriber of the DataReader.

property subscription_matched_status

Get the SubscriptionMatchedStatus for the DataReader.

take (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader) → list

Take copies of all available samples (data and info).

take_data (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader) → list

Take copies of all available valid data.

take_loaned (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader) →
rti.connextdds.TopicBuiltinTopicData.LoanedSamples

Take all available samples (data and info) and return them in a loaned container.

property topic_description

Returns the TopicDescription associated with the DataReader.

property topic_name

Get the topic name associated with this DataReader.

property type_name

Get the type name associated with this DataReader.

wait_for_historical_data (*self*: rti.connextdds.TopicBuiltinTopicData.DataReader,
max_wait: rti.connextdds.Duration) → None

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind.

wait_for_historical_data_async (*self*: rti.connextdds.TopicBuiltinTopic-
 Data.DataReader, *max_wait*:
 rti.connextdds.Duration) → object

Waits until all “historical” data is received for DataReaders that have a non-VOLATILE Durability Qos kind. This call is awaitable and only for use with asyncio.

class DataReaderListener

Bases: pybind11_object

__init__ (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderListener) → None**__module__** = 'rti.connexdds'**on_data_available** (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader) → None

Data available callback.

on_liveliness_changed (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.LivelinessChangedStatus) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SampleLostStatus) → None

Sample lost callback.

on_sample_rejected (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class DataReaderSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) → *rti.connexdds.TopicBuiltinTopicData.DataReaderSeq*

__bool__ (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, *x*: rti.connexdds.TopicBuiltinTopicData.DataReader) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, *s*: slice) -> rti.connexdds.TopicBuiltinTopicData.DataReaderSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, *arg0*: int) -> rti.connexdds.TopicBuiltinTopicData.DataReader

__hash__ = None

__iadd__ (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) → *rti.connexdds.TopicBuiltinTopicData.DataReaderSeq*

__imul__ (*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, *arg0*: int) → *rti.connexdds.TopicBuiltinTopicData.DataReaderSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) -> None

2. **__init__**(*self*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) -> None

Copy constructor

3. `__init__(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) -> Iterator`

`__len__(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) -> int`

`__module__ = 'rti.connexdds'`

`__mul__(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, arg0: int) -> rti.connexdds.TopicBuiltinTopicData.DataReaderSeq`

`__ne__(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, arg0: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) -> bool`

`__rmul__(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, arg0: int) -> rti.connexdds.TopicBuiltinTopicData.DataReaderSeq`

`__setitem__(*args, **kwargs)`
 Overloaded function.
 1. `__setitem__(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, arg0: int, arg1: rti.connexdds.TopicBuiltinTopicData.DataReader) -> None`
 2. `__setitem__(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, arg0: slice, arg1: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) -> None`
 Assign list elements using a slice object

`append(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, x: rti.connexdds.TopicBuiltinTopicData.DataReader) -> None`
 Add an item to the end of the list

`clear(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) -> None`
 Clear the contents

`count(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, x: rti.connexdds.TopicBuiltinTopicData.DataReader) -> int`
 Return the number of times `x` appears in the list

`extend(*args, **kwargs)`
 Overloaded function.
 1. `extend(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, L: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq) -> None`
 Extend the list by appending all the items in the given list
 2. `extend(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, L: Iterable) -> None`
 Extend the list by appending all the items in the given list

`insert(self: rti.connexdds.TopicBuiltinTopicData.DataReaderSeq, i: int, x: rti.connexdds.TopicBuiltinTopicData.DataReader) -> None`
 Insert an item at a given position.

`pop(*args, **kwargs)`
 Overloaded function.

1. `pop(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq) -> rti.connextdds.TopicBuiltinTopicData.DataReader`

Remove and return the last item

2. `pop(self: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, i: int) -> rti.connextdds.TopicBuiltinTopicData.DataReader`

Remove and return the item at index `i`

remove (*self*: rti.connextdds.TopicBuiltinTopicData.DataReaderSeq, *x*: rti.connextdds.TopicBuiltinTopicData.DataReader) → None

Remove the first item from the list whose value is `x`. It is an error if there is no such item.

class DataWriter

Bases: *IEntity, IAnyDataWriter*

__enter__ (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter) → *rti.connextdds.TopicBuiltinTopicData.DataWriter*

Enter a context for this DataWriter, to be cleaned up on exiting context

__eq__ (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg0*: rti.connextdds.TopicBuiltinTopicData.DataWriter) → bool

Test for equality.

__exit__ (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg0*: object, *arg1*: object, *arg2*: object) → None

Exit the context for this DataWriter, cleaning up resources.

__hash__ = None

__init__ (**args, **kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, topic: rti.connextdds.TopicBuiltinTopicData.Topic) -> None`

Creates a DataWriter in the implicit publisher with default QoS.

2. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, topic: rti.connextdds.TopicBuiltinTopicData.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.TopicBuiltinTopicData.DataWriterListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Creates a DataWriter in the implicit publisher with specific QoS and optionally a listener.

3. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.TopicBuiltinTopicData.Topic) -> None`

Creates a DataWriter in a publisher with default QoS.

4. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, pub: rti.connextdds.Publisher, topic: rti.connextdds.TopicBuiltinTopicData.Topic, qos: rti.connextdds.DataWriterQos, listener: rti.connextdds.TopicBuiltinTopicData.DataWriterListener = None, mask: rti.connextdds.StatusMask = StatusMask.ALL) -> None`

Creates a DataWriter in a publisher with specific QoS and optionally a listener.

5. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, writer: rti.connextdds.IAnyDataWriter) -> None`

Create a typed DataWriter from an AnyDataWriter.

6. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, entity: rti.connextdds.IEntity) -> None`

Create a typed DataWriter from an Entity.

`__lshift__ (*args, **kwargs)`

Overloaded function.

1. `__lshift__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, arg0: rti.connextdds.DataWriterQos) -> rti.connextdds.TopicBuiltinTopicData.DataWriter`

Sets the DataWriterQos.

2. `__lshift__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, arg0: Tuple[rti.connextdds.TopicBuiltinTopicData, rti.connextdds.Time]) -> rti.connextdds.TopicBuiltinTopicData.DataWriter`

Writes a paired sample with a timestamp.

3. `__lshift__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, arg0: Tuple[rti.connextdds.TopicBuiltinTopicData, rti.connextdds.InstanceHandle]) -> rti.connextdds.TopicBuiltinTopicData.DataWriter`

Writes a paired sample with an instance handle.

4. `__lshift__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, arg0: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) -> rti.connextdds.TopicBuiltinTopicData.DataWriter`

Writes a sequence of pairs of samples with timestamps.

5. `__lshift__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, arg0: rti.connextdds.TopicBuiltinTopicDataSeq) -> rti.connextdds.TopicBuiltinTopicData.DataWriter`

Writes a sequence of samples.

6. `__lshift__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, arg0: rti.connextdds.TopicBuiltinTopicData) -> rti.connextdds.TopicBuiltinTopicData.DataWriter`

Writes a sample.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, arg0: rti.connextdds.TopicBuiltinTopicData.DataWriter) -> bool`

Test for inequality.

`__rshift__(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, arg0: rti.connextdds.DataWriterQos) -> rti.connextdds.TopicBuiltinTopicData.DataWriter`

Get the DataWriterQos.

`assert_liveliness(self: rti.connextdds.TopicBuiltinTopicData.DataWriter) -> None`

Manually asserts the liveliness of the DataWriter.

`close(self: rti.connextdds.TopicBuiltinTopicData.DataWriter) -> None`

Close this DataWriter.

`create_data(self: rti.connextdds.TopicBuiltinTopicData.DataWriter) -> rti.connextdds.TopicBuiltinTopicData`

Create data of the writer's associated type and initialize it.

property datawriter_cache_status

Get a copy of the cache status for this writer.

property datawriter_protocol_status

Get a copy of the protocol status for this writer.

dispose_instance (*args, **kwargs)

Overloaded function.

1. `dispose_instance(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> rti.connextdds.TopicBuiltinTopicData.DataWriter`

Dispose an instance.

2. `dispose_instance(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> rti.connextdds.TopicBuiltinTopicData.DataWriter`

Dispose an instance with a timestamp.

3. `dispose_instance(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> None`

Dispose an instance with params.

dispose_instance_async (*args, **kwargs)

Overloaded function.

1. `dispose_instance_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle) -> object`

Dispose an instance.

2. `dispose_instance_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Dispose an instance with a timestamp.

3. `dispose_instance_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, key_holder: rti.connextdds.TopicBuiltinTopicData) -> object`

Dispose the instance associated with key_holder.

4. `dispose_instance_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, key_holder: rti.connextdds.TopicBuiltinTopicData, timestamp: rti.connextdds.Time) -> object`

Dispose the instance associated with key_holder using a timestamp

5. `dispose_instance_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, params: rti.connextdds.WriteParams) -> object`

Dispose an instance with params.

static find_all_by_topic (publisher: rti.connextdds.Publisher, topic_name: str) → rti.connextdds.TopicBuiltinTopicData.DataWriterSeq

Retrieve all DataWriters for the given topic name in the publisher.

static find_by_name (*args, **kwargs)

Overloaded function.

1. `find_by_name(participant: rti.connextdds.DomainParticipant, name: str) -> Optional[rti.connextdds.TopicBuiltinTopicData.DataWriter]`

Find DataWriter in DomainParticipant with the provided name, returning the first found.

2. `find_by_name(publisher: rti.connextdds.Publisher, name: str) -> Optional[rti.connextdds.TopicBuiltinTopicData.DataWriter]`

Find DataWriter in Publisher with the DataReader's name, returning the first found.

static find_by_topic (*publisher*: rti.connextdds.Publisher, *name*: str) → Optional[*rti.connextdds.TopicBuiltinTopicData.DataWriter*]

Find DataWriter in publisher with a topic name, returning the first found.

flush (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter) → None

Flushes the batch in progress in the context of the calling thread.

is_matched_subscription_active (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg0*: rti.connextdds.InstanceHandle) → bool

A boolean indicating whether or not the matched subscription is active.

is_sample_app_acknowledged (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *sample_id*: rti.connextdds.SampleIdentity) → bool

Indicates if a sample is considered application-acknowledged.

key_value (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.TopicBuiltinTopicData*

Retrieve the instance key that corresponds to an instance handle.

property listener

Get the listener associated with the DataWriter or set the listener.

property liveliness_lost_status

Get a copy of the LivelinessLostStatus.

lookup_instance (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *key_holder*: rti.connextdds.TopicBuiltinTopicData) → *rti.connextdds.InstanceHandle*

Retrieve the instance handle that corresponds to an instance key_holder

matched_subscription_data (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.SubscriptionBuiltinTopicData*

Get the SubscriptionBuiltinTopicData for a subscription matched to this DataWriter.

matched_subscription_datawriter_protocol_status (**args*, ***kwargs*)

Overloaded function.

1. **matched_subscription_datawriter_protocol_status**(*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *handle*: rti.connextdds.InstanceHandle) -> rti.connextdds.DataWriterProtocolStatus

Get a copy of the protocol status for this writer per a matched subscription handle.

2. **matched_subscription_datawriter_protocol_status**(*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *locator*: rti.connextdds.Locator) -> rti.connextdds.DataWriterProtocolStatus

Get a copy of the protocol status for this writer per a matched subscription locator.

matched_subscription_participant_data (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *handle*: rti.connextdds.InstanceHandle) → *rti.connextdds.ParticipantBuiltinTopicData*

Get the ParticipantBuiltinTopicData for a subscription matched to this DataWriter.

property matched_subscriptions

Get a copy of the list of the currently matched subscription handles.

property matched_subscriptions_locators

The locators used to communicate with matched DataReaders.

property offered_deadline_missed_status

Get a copy of the OfferedDeadlineMissedStatus.

property offered_incompatible_qos_status

Get a copy of the OfferedIncompatibleQosStatus

property publication_matched_status

Get a copy of the PublicationMatchedStatus

property publisher

Get the Publisher that owns this DataWriter.

property qos

The DataWriterQos for this DataWriter. This property's getter returns a deep copy.

register_instance (**args*, ***kwargs*)

Overloaded function.

1. register_instance(*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *key_holder*: rti.connextdds.TopicBuiltinTopicData) → rti.connextdds.InstanceHandle

Informs RTI Connex that the application will be modifying a particular instance.

2. register_instance(*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *key_holder*: rti.connextdds.TopicBuiltinTopicData, *timestamp*: rti.connextdds.Time) → rti.connextdds.InstanceHandle

Informs RTI Connex that the application will be modifying a particular instance and specified the timestamp.

3. register_instance(*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *key_holder*: rti.connextdds.TopicBuiltinTopicData, *params*: rti.connextdds.WriteParams) → rti.connextdds.InstanceHandle

Registers instance with parameters.

property reliable_reader_activity_changed_status

Get a copy of the reliable reader activity changed status for this writer.

property reliable_writer_cache_changed_status

Get a copy of the reliable cache status for this writer.

property service_request_accepted_status

Get a copy of the service request accepted status for this writer.

set_listener (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *listener*: rti.connexdds.TopicBuiltinTopicData.DataWriterListener, *event_mask*: rti.connexdds.StatusMask) → None

Set the listener and event mask for the DataWriter.

property topic

Get the Topic object associated with this DataWriter.

property topic_name

Get the topic name associated with this DataWriter.

property type_name

Get the type name for the topic object associated with this DataWriter.

unregister_instance (**args*, ***kwargs*)

Overloaded function.

1. `unregister_instance(self: rti.connexdds.TopicBuiltinTopicData.DataWriter, handle: rti.connexdds.InstanceHandle) -> rti.connexdds.TopicBuiltinTopicData.DataWriter`

Unregister an instance.

2. `unregister_instance(self: rti.connexdds.TopicBuiltinTopicData.DataWriter, handle: rti.connexdds.InstanceHandle, timestamp: rti.connexdds.Time) -> rti.connexdds.TopicBuiltinTopicData.DataWriter`

Unregister an instance with timestamp.

3. `unregister_instance(self: rti.connexdds.TopicBuiltinTopicData.DataWriter, params: rti.connexdds.WriteParams) -> None`

Unregister an instance with parameters.

unregister_instance_async (**args*, ***kwargs*)

Overloaded function.

1. `unregister_instance_async(self: rti.connexdds.TopicBuiltinTopicData.DataWriter, handle: rti.connexdds.InstanceHandle) -> object`

Unregister an instance.

2. `unregister_instance_async(self: rti.connexdds.TopicBuiltinTopicData.DataWriter, handle: rti.connexdds.InstanceHandle, timestamp: rti.connexdds.Time) -> object`

Unregister an instance with timestamp.

3. `unregister_instance_async(self: rti.connexdds.TopicBuiltinTopicData.DataWriter, key_holder: rti.connexdds.TopicBuiltinTopicData) -> object`

Unregister the instance associated with `key_holder`.

4. `unregister_instance_async(self: rti.connexdds.TopicBuiltinTopicData.DataWriter, key_holder: rti.connexdds.TopicBuiltinTopicData, timestamp: rti.connexdds.Time) -> object`

Unregister the instance associate with `key_holder` using a timestamp.

5. `unregister_instance_async(self: rti.connexdds.TopicBuiltinTopicData.DataWriter, params: rti.connexdds.WriteParams) -> object`

Unregister an instance with parameters.

wait_for_acknowledgments (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *max_wait*: rti.connextdds.Duration) → None

Blocks the calling thread until all data written by a reliable DataWriter is acknowledged or until the timeout expires.

wait_for_asynchronous_publishing (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *max_wait*: rti.connextdds.Duration) → None

This operation blocks the calling thread (up to *max_wait*) until all data written by the asynchronous DataWriter is sent and acknowledged (if reliable) by all matched DataReader entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that *max_wait* elapsed before all the data was sent and/or acknowledged.

In other words, this guarantees that sending to best effort DataReader is complete in addition to what DataWriter.wait_for_acknowledgments() provides.

If the DataWriter does not have PublishMode kind set to PublishModeKind.ASYNCHRONOUS the operation will complete immediately

wait_for_asynchronous_publishing_async (*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *max_wait*: rti.connextdds.Duration) → object

This function is awaitable until either a timeout of *max_wait* or all data written by the asynchronous DataWriter is sent and acknowledged (if reliable) by all matched DataReader entities. A successful completion indicates that all the samples written have been sent and acknowledged where applicable; a time out indicates that *max_wait* elapsed before all the data was sent and/or acknowledged. This function works with asyncio.

In other words, this guarantees that sending to best effort DataReader is complete in addition to what DataWriter.wait_for_acknowledgments() provides.

If the DataWriter does not have PublishMode kind set to PublishModeKind.ASYNCHRONOUS the operation will complete immediately

write (**args*, ***kwargs*)

Overloaded function.

1. write(*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *samples*: rti.connextdds.TopicBuiltinTopicDataSeq) → None

Write a sequence of samples.

2. write(*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *samples*: rti.connextdds.TopicBuiltinTopicDataSeq, *timestamp*: rti.connextdds.Time) → None

Write a sequence of samples with a timestamp.

3. write(*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *sample*: rti.connextdds.TopicBuiltinTopicData) → None

Write a sample.

4. write(*self*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *sample*: rti.connextdds.TopicBuiltinTopicData, *timestamp*: rti.connextdds.Time) → None

Write a sample with a specified timestamp.

5. `write(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, sample: rti.connextdds.TopicBuiltinTopicData, handle: rti.connextdds.InstanceHandle) -> None`

Write a sample with an instance handle.

6. `write(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, sample: rti.connextdds.TopicBuiltinTopicData, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> None`

Write a sample with an instance handle and specified timestamp.

7. `write(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, sample: rti.connextdds.TopicBuiltinTopicData, params: rti.connextdds.WriteParams) -> None`

Write with advanced parameters.

write_async (*args, **kwargs)

Overloaded function.

1. `write_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, sample: rti.connextdds.TopicBuiltinTopicData) -> object`

Write a sample. This method is awaitable and is only for use with asyncio.

2. `write_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, sample: rti.connextdds.TopicBuiltinTopicData, timestamp: rti.connextdds.Time) -> object`

Write a sample with a specified timestamp. This methods is awaitable and only for use with asyncio.

3. `write_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, sample: rti.connextdds.TopicBuiltinTopicData, handle: rti.connextdds.InstanceHandle) -> object`

Write a sample with an instance handle. This method is awaitable and only for use with asyncio.

4. `write_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, sample: rti.connextdds.TopicBuiltinTopicData, handle: rti.connextdds.InstanceHandle, timestamp: rti.connextdds.Time) -> object`

Write a sample with an instance handle and specified timestamp. This method is awaitable and only for use with asyncio.

5. `write_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, samples: rti.connextdds.TopicBuiltinTopicDataSeq) -> object`

Write a sequence of samples. This method is awaitable and only for use with asyncio.

6. `write_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, samples: rti.connextdds.TopicBuiltinTopicDataSeq, timestamp: rti.connextdds.Time) -> object`

Write a sequence of samples with a timestamp. This method is awaitable and only for use with asyncio.

7. `write_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, samples: rti.connextdds.TopicBuiltinTopicDataSeq, handles: rti.connextdds.InstanceHandleSeq) -> object`

Write a sequence of samples with their instance handles. This method is awaitable and only for use with asyncio.

8. `write_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, samples: rti.connextdds.TopicBuiltinTopicDataSeq, handles: rti.connextdds.InstanceHandleSeq, timestamp: rti.connextdds.Time) -> object`

Write a sequence of samples with their instance handles and a timestamp. This method is awaitable and only for use with asyncio.

9. `write_async(self: rti.connextdds.TopicBuiltinTopicData.DataWriter, sample: rti.connextdds.TopicBuiltinTopicData, params: rti.connextdds.WriteParams) -> object`

Write with advanced parameters.

class DataWriterListener

Bases: pybind11_object

__init__ (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterListener) → None**__module__** = 'rti.connexdds'**on_application_acknowledgment** (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.AcknowledgmentInfo) → None

On application acknowledgment callback

on_instance_replaced (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.InstanceHandle) → None

On instance replaced callback.

on_liveliness_lost (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.LivelinessLostStatus) → None

Liveliness lost callback.

on_offered_deadline_missed (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

on_offered_incompatible_qos (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

on_publication_matched (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.PublicationMatchedStatus) → None

Publication matched callback.

on_reliable_reader_activity_changed (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

```
on_reliable_writer_cache_changed (self: rti.connexdds.TopicBuiltinTopic-
                                     Data.DataWriterListener, arg0:
                                     rti.connexdds.TopicBuiltinTopic-
                                     Data.DataWriter, arg1:
                                     rti.connexdds.ReliableWriterCacheChanged-
                                     Status) →
                                     None
```

Reliable writer cache changed callback.

```
on_service_request_accepted (self: rti.connexdds.TopicBuiltinTopic-
                                  Data.DataWriterListener, arg0:
                                  rti.connexdds.TopicBuiltinTopicData.DataWriter,
                                  arg1: rti.connexdds.ServiceRequestAcceptedStatus)
                                  → None
```

On service request accepted callback.

class DataWriterSeq

Bases: pybind11_object

```
__add__ (self: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, arg0:
          rti.connexdds.TopicBuiltinTopicData.DataWriterSeq) →
          rti.connexdds.TopicBuiltinTopicData.DataWriterSeq
```

```
__bool__ (self: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq) → bool
          Check whether the list is nonempty
```

```
__contains__ (self: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, x:
               rti.connexdds.TopicBuiltinTopicData.DataWriter) → bool
```

Return true the container contains *x*

```
__delitem__ (*args, **kwargs)
```

Overloaded function.

```
1. __delitem__(self: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, arg0: int) ->
   None
```

Delete the list elements at index *i*

```
2. __delitem__(self: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, arg0: slice) ->
   None
```

Delete list elements using a slice object

```
__eq__ (self: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, arg0:
         rti.connexdds.TopicBuiltinTopicData.DataWriterSeq) → bool
```

```
__getitem__ (*args, **kwargs)
```

Overloaded function.

```
1. __getitem__(self: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, s: slice) ->
   rti.connexdds.TopicBuiltinTopicData.DataWriterSeq
```

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.DataWriter`

`__hash__ = None`

`__iadd__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) -> rti.connextdds.TopicBuiltinTopicData.DataWriterSeq`

`__imul__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.DataWriterSeq`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) -> None`
2. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) -> Iterator`

`__len__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) -> int`

`__module__ = 'rti.connextdds'`

`__mul__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.DataWriterSeq`

`__ne__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) -> bool`

`__rmul__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.DataWriterSeq`

`__setitem__(*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: int, arg1: rti.connextdds.TopicBuiltinTopicData.DataWriter) -> None`
2. `__setitem__(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, arg0: slice, arg1: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) -> None`

Assign list elements using a slice object

`append(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq, x: rti.connextdds.TopicBuiltinTopicData.DataWriter) -> None`

Add an item to the end of the list

`clear(self: rti.connextdds.TopicBuiltinTopicData.DataWriterSeq) -> None`

Clear the contents

count (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, *x*: rti.connexdds.TopicBuiltinTopicData.DataWriter) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, *L*: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq) → None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, *L*: Iterable) → None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, *i*: int, *x*: rti.connexdds.TopicBuiltinTopicData.DataWriter) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq) → rti.connexdds.TopicBuiltinTopicData.DataWriter

Remove and return the last item

2. pop(*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, *i*: int) → rti.connexdds.TopicBuiltinTopicData.DataWriter

Remove and return the item at index *i*

remove (*self*: rti.connexdds.TopicBuiltinTopicData.DataWriterSeq, *x*: rti.connexdds.TopicBuiltinTopicData.DataWriter) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class ITopicDescription

Bases: *IEntity*

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

property name

The name of the entity conforming to the ITopicDescription interface.

property participant

The parent DomainParticipant.

property type_name

The name of the associated type.

class LoanedSample

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.TopicBuiltinTopicData.LoanedSample) → None

Basic constructor

2. `__init__(self: rti.connextdds.TopicBuiltinTopicData.LoanedSample, data: rti.connextdds.TopicBuiltinTopicData, info: rti.connextdds.SampleInfo) -> None`

Construct `LoanedSample` with data and info.

`__iter__(self: rti.connextdds.TopicBuiltinTopicData.LoanedSample) -> object`

`__module__ = 'rti.connextdds'`

property data

Get the data associated with the sample.

property info

Get the info associated with the sample.

class LoanedSamples

Bases: `pybind11_object`

`__enter__(self: rti.connextdds.TopicBuiltinTopicData.LoanedSamples) -> rti.connextdds.TopicBuiltinTopicData.LoanedSamples`

Enter a context for the loaned samples, loan returned on context exit.

`__exit__(self: rti.connextdds.TopicBuiltinTopicData.LoanedSamples, arg0: object, arg1: object, arg2: object) -> None`

Exit the context for the loaned samples, returning the resources.

`__getitem__(self: rti.connextdds.TopicBuiltinTopicData.LoanedSamples, arg0: int) -> rti.connextdds.TopicBuiltinTopicData.LoanedSample`

Access a `LoanedSample` object in an array-like syntax

`__init__(self: rti.connextdds.TopicBuiltinTopicData.LoanedSamples) -> None`

Create an empty `LoanedSamples` object.

`__iter__(self: rti.connextdds.TopicBuiltinTopicData.LoanedSamples) -> Iterator`

`__len__(self: rti.connextdds.TopicBuiltinTopicData.LoanedSamples) -> int`

Get the number of samples in the loan.

`__module__ = 'rti.connextdds'`

property length

Get the number of samples in the loan.

`return_loan(self: rti.connextdds.TopicBuiltinTopicData.LoanedSamples) -> None`

Returns the loan to the `DataReader`.

class NoOpDataReaderListener

Bases: `DataReaderListener`

`__init__(self: rti.connextdds.TopicBuiltinTopicData.NoOpDataReaderListener) -> None`

`__module__ = 'rti.connextdds'`

on_data_available (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader) → None

Data available callback.

on_liveliness_changed (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.LivelinessChangedStatus) → None

Liveliness changed callback.

on_requested_deadline_missed (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedDeadlineMissedStatus) → None

Requested deadline missed callback.

on_requested_incompatible_qos (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.RequestedIncompatibleQosStatus) → None

Requested incompatible QoS callback.

on_sample_lost (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SampleLostStatus) → None

Sample lost callback.

on_sample_rejected (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SampleRejectedStatus) → None

Sample rejected callback.

on_subscription_matched (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataReader, *arg1*: rti.connexdds.SubscriptionMatchedStatus) → None

Subscription matched callback.

class NoOpDataWriterListener

Bases: *DataWriterListener*

__init__ (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener) → None

`__module__ = 'rti.connexdds'`

`on_application_acknowledgment` (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.AcknowledgmentInfo) → None

On application acknowledgment callback

`on_instance_replaced` (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.InstanceHandle) → None

On instance replaced callback.

`on_liveliness_lost` (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.LivelinessLostStatus) → None

Liveliness lost callback.

`on_offered_deadline_missed` (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.OfferedDeadlineMissedStatus) → None

Offered deadline missed callback.

`on_offered_incompatible_qos` (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.OfferedIncompatibleQosStatus) → None

Offered incompatible QoS callback.

`on_publication_matched` (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.PublicationMatchedStatus) → None

Publication matched callback.

`on_reliable_reader_activity_changed` (*self*: rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connexdds.ReliableReaderActivityChangedStatus) → None

Reliable reader activity changed callback.

on_reliable_writer_cache_changed (*self*: rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.ReliableWriterCacheChangedStatus) → None

Reliable writer cache changed callback.

on_service_request_accepted (*self*: rti.connextdds.TopicBuiltinTopicData.NoOpDataWriterListener, *arg0*: rti.connextdds.TopicBuiltinTopicData.DataWriter, *arg1*: rti.connextdds.ServiceRequestAcceptedStatus) → None

On service request accepted callback.

class NoOpTopicListener

Bases: *TopicListener*

__init__ (*self*: rti.connextdds.TopicBuiltinTopicData.NoOpTopicListener) → None

__module__ = 'rti.connextdds'

on_inconsistent_topic (*self*: rti.connextdds.TopicBuiltinTopicData.NoOpTopicListener, *arg0*: rti.connextdds.TopicBuiltinTopicData.Topic, *arg1*: rti.connextdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class Sample

Bases: *pybind11_object*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.TopicBuiltinTopicData.Sample, *data*: rti.connextdds.TopicBuiltinTopicData, *info*: rti.connextdds.SampleInfo) -> None

Construct Sample with data and info.

2. **__init__**(*self*: rti.connextdds.TopicBuiltinTopicData.Sample, *sample*: rti.connextdds.TopicBuiltinTopicData.Sample) -> None

Copy constructor.

3. **__init__**(*self*: rti.connextdds.TopicBuiltinTopicData.Sample) -> None

Basic constructor

4. **__init__**(*self*: rti.connextdds.TopicBuiltinTopicData.Sample, *loaned_sample*: rti.connextdds.TopicBuiltinTopicData.LoanedSample) -> None

Construct a sample with a loaned sample.

__iter__ (*self*: rti.connextdds.TopicBuiltinTopicData.Sample) → object

__module__ = 'rti.connextdds'

property data

The data associated with the sample.

property info

Get the info associated with the sample.

class SharedSamples

Bases: `pybind11_object`

__getitem__ (*self*: `rti.connexdds.TopicBuiltinTopicData.SharedSamples`, *arg0*: `int`) → `rti.connexdds.TopicBuiltinTopicData.LoanedSample`

Get the sample at the specified index.

__init__ (*self*: `rti.connexdds.TopicBuiltinTopicData.SharedSamples`, *loaned_samples*: `rti.connexdds.TopicBuiltinTopicData.LoanedSamples`) → `None`

Constructs an instance of `SharedSamples`, removing ownership of the loan from the `LoanedSamples`.

__iter__ (*self*: `rti.connexdds.TopicBuiltinTopicData.SharedSamples`) → `Iterator`

Make a sample iterator

__len__ (*self*: `rti.connexdds.TopicBuiltinTopicData.SharedSamples`) → `int`

Returns the number of samples.

__module__ = `'rti.connexdds'`

class Topic

Bases: `ITopicDescription`, `IAnyTopic`

__eq__ (*self*: `rti.connexdds.TopicBuiltinTopicData.Topic`, *arg0*: `rti.connexdds.TopicBuiltinTopicData.Topic`) → `bool`

Test for equality.

__hash__ = `None`

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.TopicBuiltinTopicData.Topic, entity: rti.connexdds.IEntity) -> None`

Cast an Entity to a Topic.

2. `__init__(self: rti.connexdds.TopicBuiltinTopicData.Topic, topic_description: rti.connexdds.TopicBuiltinTopicData.ITopicDescription) -> None`

Cast an ITopicDescription to a Topic.

3. `__init__(self: rti.connexdds.TopicBuiltinTopicData.Topic, topic: rti.connexdds.IAnyTopic) -> None`

Create a typed Topic from an AnyTopic.

__module__ = `'rti.connexdds'`

__ne__ (*self*: rti.connexdds.TopicBuiltinTopicData.Topic, *arg0*: rti.connexdds.TopicBuiltinTopicData.Topic) → bool

Test for inequality.

static find (*participant*: rti.connexdds.DomainParticipant, *name*: str) → Optional[rti.connexdds.TopicBuiltinTopicData.Topic]

Look up a Topic by its name in the DomainParticipant.

property inconsistent_topic_status

Get a copy of the current InconsistentTopicStatus for this Topic.

property listener

The listener.

property qos

Get the TopicQos for this Topic.

This property's getter returns a deep copy.

set_listener (*self*: rti.connexdds.TopicBuiltinTopicData.Topic, *listener*: rti.connexdds.TopicBuiltinTopicData.TopicListener, *event_mask*: rti.connexdds.StatusMask) → None

Set the listener and event mask.

class TopicDescription

Bases: *ITopicDescription*

__eq__ (*self*: rti.connexdds.TopicBuiltinTopicData.TopicDescription, *arg0*: rti.connexdds.TopicBuiltinTopicData.TopicDescription) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.TopicBuiltinTopicData.TopicDescription, *topic*: rti.connexdds.TopicBuiltinTopicData.ITopicDescription) -> None

Cast a Topic to a TopicDescription.

2. **__init__**(*self*: rti.connexdds.TopicBuiltinTopicData.TopicDescription, *entity*: rti.connexdds.IEntity) -> None

Cast a Topic to a TopicDescription.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.TopicBuiltinTopicData.TopicDescription, *arg0*: rti.connexdds.TopicBuiltinTopicData.TopicDescription) → bool

Test for inequality.

class TopicListener

Bases: *pybind11_object*

`__init__` (*self*: rti.connexdds.TopicBuiltinTopicData.TopicListener) → None

`__module__` = 'rti.connexdds'

`on_inconsistent_topic` (*self*: rti.connexdds.TopicBuiltinTopicData.TopicListener, *arg0*: rti.connexdds.TopicBuiltinTopicData.Topic, *arg1*: rti.connexdds.InconsistentTopicStatus) → None

Inconsistent topic callback.

class TopicSeq

Bases: pybind11_object

`__add__` (*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *arg0*: rti.connexdds.TopicBuiltinTopicData.TopicSeq) → *rti.connexdds.TopicBuiltinTopicData.TopicSeq*

`__bool__` (*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq) → bool
Check whether the list is nonempty

`__contains__` (*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *x*: rti.connexdds.TopicBuiltinTopicData.Topic) → bool

Return true the container contains *x*

`__delitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__delitem__`(*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. `__delitem__`(*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *arg0*: slice) -> None

Delete list elements using a slice object

`__eq__` (*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *arg0*: rti.connexdds.TopicBuiltinTopicData.TopicSeq) → bool

`__getitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__`(*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *s*: slice) -> rti.connexdds.TopicBuiltinTopicData.TopicSeq

Retrieve list elements using a slice object

2. `__getitem__`(*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *arg0*: int) -> rti.connexdds.TopicBuiltinTopicData.Topic

`__hash__` = None

`__iadd__` (*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *arg0*: rti.connexdds.TopicBuiltinTopicData.TopicSeq) → *rti.connexdds.TopicBuiltinTopicData.TopicSeq*

`__imul__` (*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *arg0*: int) → *rti.connexdds.TopicBuiltinTopicData.TopicSeq*

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.TopicBuiltinTopicData.TopicSeq) -> None`
2. `__init__(self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, arg0: rti.connexdds.TopicBuiltinTopicData.TopicSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, arg0: Iterable) -> None`

__iter__ (self: rti.connexdds.TopicBuiltinTopicData.TopicSeq) → Iterator

__len__ (self: rti.connexdds.TopicBuiltinTopicData.TopicSeq) → int

__module__ = 'rti.connexdds'

__mul__ (self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, arg0: int) → rti.connexdds.TopicBuiltinTopicData.TopicSeq

__ne__ (self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, arg0: rti.connexdds.TopicBuiltinTopicData.TopicSeq) → bool

__rmul__ (self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, arg0: int) → rti.connexdds.TopicBuiltinTopicData.TopicSeq

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, arg0: int, arg1: rti.connexdds.TopicBuiltinTopicData.Topic) -> None`
2. `__setitem__(self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, arg0: slice, arg1: rti.connexdds.TopicBuiltinTopicData.TopicSeq) -> None`

Assign list elements using a slice object

append (self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, x: rti.connexdds.TopicBuiltinTopicData.Topic) → None

Add an item to the end of the list

clear (self: rti.connexdds.TopicBuiltinTopicData.TopicSeq) → None

Clear the contents

count (self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, x: rti.connexdds.TopicBuiltinTopicData.Topic) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, L: rti.connexdds.TopicBuiltinTopicData.TopicSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.TopicBuiltinTopicData.TopicSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *i*: int, *x*: rti.connexdds.TopicBuiltinTopicData.Topic) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq) -> rti.connexdds.TopicBuiltinTopicData.Topic

Remove and return the last item

2. pop(*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *i*: int) -> rti.connexdds.TopicBuiltinTopicData.Topic

Remove and return the item at index *i*

remove (*self*: rti.connexdds.TopicBuiltinTopicData.TopicSeq, *x*: rti.connexdds.TopicBuiltinTopicData.Topic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class ValidLoanedSamples

Bases: pybind11_object

__enter__ (*self*: rti.connexdds.TopicBuiltinTopicData.ValidLoanedSamples) → rti.connexdds.TopicBuiltinTopicData.ValidLoanedSamples

__exit__ (*self*: rti.connexdds.TopicBuiltinTopicData.ValidLoanedSamples, *arg0*: object, *arg1*: object, *arg2*: object) → None

__init__ (**args*, ***kwargs*)

__iter__ (*self*: rti.connexdds.TopicBuiltinTopicData.ValidLoanedSamples) → Iterator

__module__ = 'rti.connexdds'

class WriterContentFilter

Bases: ContentFilter

__init__ (*self*: rti.connexdds.TopicBuiltinTopicData.WriterContentFilter) → None

__module__ = 'rti.connexdds'

writer_attach (*self*: rti.connexdds.TopicBuiltinTopicData.WriterContentFilter) → Optional[object]

A writer-side filtering API to create some state that can facilitate filtering on the writer side.

writer_compile (*self*: rti.connexdds.TopicBuiltinTopicData.WriterContentFilter, *writer_filter_data*: Optional[object], *property*: rti.connexdds.ExpressionProperty, *expression*: str, *parameters*: rti.connexdds.StringSeq, *type_code*: Optional[rti.connexdds.DynamicType], *type_class_name*: str, *cookie*: rti.connexdds.Cookie) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching DataReader.

writer_detach (*self*: rti.connextdds.TopicBuiltinTopicData.WriterContentFilter, *writer_filter_data*: *Optional[object]*) → None

A writer-side filtering API to clean up a previously created state using `writer_attach`.

writer_evaluate (*self*: rti.connextdds.TopicBuiltinTopicData.WriterContentFilter, *writer_filter_data*: *Optional[object]*, *sample*: rti.connextdds.TopicBuiltinTopicData, *meta_data*: rti.connextdds.FilterSampleInfo) → *rti.connextdds.CookieVector*

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching `DataReader`.

writer_finalize (*self*: rti.connextdds.TopicBuiltinTopicData.WriterContentFilter, *writer_filter_data*: *Optional[object]*, *cookie*: rti.connextdds.Cookie) → None

A writer-side filtering API to clean up a previously compiled instance of the content filter.

writer_return_loan (*self*: rti.connextdds.TopicBuiltinTopicData.WriterContentFilter, *writer_filter_data*: *Optional[object]*, *cookies*: rti.connextdds.CookieVector) → None

A writer-side filtering API to return the loan on the list of `DataReaders` returned by `writer_evaluate`.

class WriterContentFilterHelper

Bases: *WriterContentFilter*

__init__ (*self*: rti.connextdds.TopicBuiltinTopicData.WriterContentFilterHelper) → None

__module__ = 'rti.connextdds'

add_cookie (*self*: rti.connextdds.TopicBuiltinTopicData.WriterContentFilterHelper, *cookie*: rti.connextdds.Cookie) → None

A helper function which will add a `Cookie` to the `Cookie` sequence that is then returned by the `writer_evaluate` function.

writer_evaluate_helper (*self*: rti.connextdds.TopicBuiltinTopicData.WriterContentFilterHelper, *writer_filter_data*: *Optional[object]*, *sample*: rti.connextdds.TopicBuiltinTopicData, *meta_data*: rti.connextdds.FilterSampleInfo) → None

A writer-side filtering API to compile an instance of the content filter according to the filter expression and parameters specified by a matching `DataReader`.

__eq__ (*self*: rti.connextdds.TopicBuiltinTopicData, *arg0*: rti.connextdds.TopicBuiltinTopicData) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connextdds.TopicBuiltinTopicData) → None

Create a default `TopicBuiltinTopicData`.

`__module__ = 'rti.connexdds'`

`__ne__ (self: rti.connexdds.TopicBuiltinTopicData, arg0: rti.connexdds.TopicBuiltinTopicData)
→ bool`

Test for inequality.

property deadline

Get the Deadline policy of the corresponding Topic.

property destination_order

Get the DestinationOrder policy of the corresponding Topic.

property durability

Get the Durability policy of the corresponding Topic.

property durability_service

Get the DurabilityService policy of the corresponding Topic.

property history

Get the History policy of the corresponding Topic.

property key

Get the DCPS key to distinguish entries.

property latency_budget

Get the LatencyBudget policy of the corresponding Topic.

property lifespan

Get the Lifespan policy of the corresponding Topic.

property liveliness

Get the Liveliness policy of the corresponding Topic.

property name

Get the name of the Topic.

property ownership

Get the Ownership policy of the corresponding Topic.

property reliability

Get the Reliability policy of the corresponding Topic.

property representation

Get the DataRepresentation policy of the corresponding Topic.

property resource_limits

Get the ResourceLimits policy of the corresponding Topic.

property topic_data

Get the TopicData policy of the corresponding Topic.

`topic_name = 'DCPSTopic'`

property transport_priority

Get the TransportPriority policy of the corresponding Topic.

property type_name

Get the name of the type attached to the Topic.

class `rti.connexdds.TopicBuiltinTopicDataSeq`

Bases: `pybind11_object`

__add__ (*self*: `rti.connexdds.TopicBuiltinTopicDataSeq`, *arg0*: `rti.connexdds.TopicBuiltinTopicDataSeq`) → *rti.connexdds.TopicBuiltinTopicDataSeq*

__bool__ (*self*: `rti.connexdds.TopicBuiltinTopicDataSeq`) → `bool`

Check whether the list is nonempty

__contains__ (*self*: `rti.connexdds.TopicBuiltinTopicDataSeq`, *x*: `rti.connexdds.TopicBuiltinTopicData`) → `bool`

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: `rti.connexdds.TopicBuiltinTopicDataSeq`, *arg0*: `int`) -> `None`

Delete the list elements at index *i*

2. **__delitem__**(*self*: `rti.connexdds.TopicBuiltinTopicDataSeq`, *arg0*: `slice`) -> `None`

Delete list elements using a slice object

__eq__ (*self*: `rti.connexdds.TopicBuiltinTopicDataSeq`, *arg0*: `rti.connexdds.TopicBuiltinTopicDataSeq`) → `bool`

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: `rti.connexdds.TopicBuiltinTopicDataSeq`, *s*: `slice`) -> `rti.connexdds.TopicBuiltinTopicDataSeq`

Retrieve list elements using a slice object

2. **__getitem__**(*self*: `rti.connexdds.TopicBuiltinTopicDataSeq`, *arg0*: `int`) -> `rti.connexdds.TopicBuiltinTopicData`

__hash__ = `None`

__iadd__ (*self*: `rti.connexdds.TopicBuiltinTopicDataSeq`, *arg0*: `rti.connexdds.TopicBuiltinTopicDataSeq`) → *rti.connexdds.TopicBuiltinTopicDataSeq*

__imul__ (*self*: `rti.connexdds.TopicBuiltinTopicDataSeq`, *arg0*: `int`) → *rti.connexdds.TopicBuiltinTopicDataSeq*

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.TopicBuiltinTopicDataSeq) -> None
2. **__init__**(self: rti.connexdds.TopicBuiltinTopicDataSeq, arg0: rti.connexdds.TopicBuiltinTopicDataSeq) -> None

Copy constructor

3. **__init__**(self: rti.connexdds.TopicBuiltinTopicDataSeq, arg0: Iterable) -> None

__iter__ (self: rti.connexdds.TopicBuiltinTopicDataSeq) → Iterator

__len__ (self: rti.connexdds.TopicBuiltinTopicDataSeq) → int

__module__ = 'rti.connexdds'

__mul__ (self: rti.connexdds.TopicBuiltinTopicDataSeq, arg0: int) →
rti.connexdds.TopicBuiltinTopicDataSeq

__ne__ (self: rti.connexdds.TopicBuiltinTopicDataSeq, arg0:
rti.connexdds.TopicBuiltinTopicDataSeq) → bool

__rmul__ (self: rti.connexdds.TopicBuiltinTopicDataSeq, arg0: int) →
rti.connexdds.TopicBuiltinTopicDataSeq

__setitem__ (*args, **kwargs)

Overloaded function.

1. **__setitem__**(self: rti.connexdds.TopicBuiltinTopicDataSeq, arg0: int, arg1: rti.connexdds.TopicBuiltinTopicData) -> None
2. **__setitem__**(self: rti.connexdds.TopicBuiltinTopicDataSeq, arg0: slice, arg1: rti.connexdds.TopicBuiltinTopicDataSeq) -> None

Assign list elements using a slice object

append (self: rti.connexdds.TopicBuiltinTopicDataSeq, x: rti.connexdds.TopicBuiltinTopicData) → None

Add an item to the end of the list

clear (self: rti.connexdds.TopicBuiltinTopicDataSeq) → None

Clear the contents

count (self: rti.connexdds.TopicBuiltinTopicDataSeq, x: rti.connexdds.TopicBuiltinTopicData) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. **extend**(self: rti.connexdds.TopicBuiltinTopicDataSeq, L: rti.connexdds.TopicBuiltinTopicDataSeq) -> None

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.TopicBuiltinTopicDataSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.TopicBuiltinTopicDataSeq, *i*: int, *x*: rti.connexdds.TopicBuiltinTopicData) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connexdds.TopicBuiltinTopicDataSeq) -> rti.connexdds.TopicBuiltinTopicData`

Remove and return the last item

2. `pop(self: rti.connexdds.TopicBuiltinTopicDataSeq, i: int) -> rti.connexdds.TopicBuiltinTopicData`

Remove and return the item at index *i*

remove (*self*: rti.connexdds.TopicBuiltinTopicDataSeq, *x*: rti.connexdds.TopicBuiltinTopicData) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.TopicBuiltinTopicDataTimestampedSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, *arg0*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq) → rti.connexdds.TopicBuiltinTopicDataTimestampedSeq

__bool__ (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, *x*: Tuple[rti.connexdds.TopicBuiltinTopicData, rti.connexdds.Time]) → bool

Return true the container contains *x*

__delitem__ (*args, **kwargs)

Overloaded function.

1. `__delitem__(self: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, *arg0*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq) → bool

`__getitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__(self: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, s: slice) -> rti.connexdds.TopicBuiltinTopicDataTimestampedSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, arg0: int) -> Tuple[rti.connexdds.TopicBuiltinTopicData, rti.connexdds.Time]`

`__hash__` = None

`__iadd__` (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, *arg0*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq) → *rti.connexdds.TopicBuiltinTopicDataTimestampedSeq*

`__imul__` (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, *arg0*: int) → *rti.connexdds.TopicBuiltinTopicDataTimestampedSeq*

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq) -> None`
2. `__init__(self: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, arg0: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, arg0: Iterable) -> None`

`__iter__` (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq) → Iterator

`__len__` (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq) → int

`__module__` = 'rti.connexdds'

`__mul__` (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, *arg0*: int) → *rti.connexdds.TopicBuiltinTopicDataTimestampedSeq*

`__ne__` (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, *arg0*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq) → bool

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` = <capsule object NULL>

`__rmul__` (*self*: rti.connexdds.TopicBuiltinTopicDataTimestampedSeq, *arg0*: int) → *rti.connexdds.TopicBuiltinTopicDataTimestampedSeq*

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, arg0: int, arg1: Tuple[rti.connextdds.TopicBuiltinTopicData, rti.connextdds.Time]) -> None`
2. `__setitem__(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, arg0: slice, arg1: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) -> None`

Assign list elements using a slice object

append (self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, x: Tuple[rti.connextdds.TopicBuiltinTopicData, rti.connextdds.Time]) → None

Add an item to the end of the list

clear (self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) → None

Clear the contents

count (self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, x: Tuple[rti.connextdds.TopicBuiltinTopicData, rti.connextdds.Time]) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, L: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, i: int, x: Tuple[rti.connextdds.TopicBuiltinTopicData, rti.connextdds.Time]) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq) -> Tuple[rti.connextdds.TopicBuiltinTopicData, rti.connextdds.Time]`

Remove and return the last item

2. `pop(self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, i: int) -> Tuple[rti.connextdds.TopicBuiltinTopicData, rti.connextdds.Time]`

Remove and return the item at index i

remove (self: rti.connextdds.TopicBuiltinTopicDataTimestampedSeq, x: Tuple[rti.connextdds.TopicBuiltinTopicData, rti.connextdds.Time]) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

```
class rti.connexdds.TopicData
```

```
Bases: pybind11_object
```

```
__eq__ (self: rti.connexdds.TopicData, arg0: rti.connexdds.TopicData) → bool
```

```
Test for equality.
```

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

```
Overloaded function.
```

```
1. __init__(self: rti.connexdds.TopicData) -> None
```

```
Create an empty TopicData QoS policy.
```

```
2. __init__(self: rti.connexdds.TopicData, data: rti.connexdds.Uint8Seq) -> None
```

```
Create a TopicData object from a data sequence.
```

```
__iter__ (self: rti.connexdds.TopicData) → Iterator
```

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.TopicData, arg0: rti.connexdds.TopicData) → bool
```

```
Test for inequality.
```

```
property value
```

```
Get/set a copy of the TopicData value.
```

```
class rti.connexdds.TopicDescription
```

```
Bases: ITopicDescription
```

```
__eq__ (self: rti.connexdds.TopicDescription, arg0: rti.connexdds.TopicDescription) → bool
```

```
Test for equality.
```

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

```
Overloaded function.
```

```
1. __init__(self: rti.connexdds.TopicDescription, topic: rti.connexdds.ITopicDescription) -> None
```

```
Cast a Topic to a TopicDescription.
```

```
2. __init__(self: rti.connexdds.TopicDescription, entity: rti.connexdds.IEntity) -> None
```

```
Cast a Topic to a TopicDescription.
```

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.TopicDescription, arg0: rti.connexdds.TopicDescription) → bool
```

```
Test for inequality.
```

class `rti.connexdds.TopicListener`Bases: `pybind11_object``__init__` (*self*: `rti.connexdds.TopicListener`) → None`__module__` = `'rti.connexdds'``on_inconsistent_topic` (*self*: `rti.connexdds.TopicListener`, *arg0*: `rti.connexdds.Topic`, *arg1*: `rti.connexdds.InconsistentTopicStatus`) → None

Inconsistent topic callback.

class `rti.connexdds.TopicQos`Bases: `pybind11_object``__eq__` (*self*: `rti.connexdds.TopicQos`, *arg0*: `rti.connexdds.TopicQos`) → bool

Test for equality

`__hash__` = None`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.TopicQos) -> None`

Create a TopicQos with the default value for each policy.

2. `__init__(self: rti.connexdds.TopicQos, topic: rti.connexdds.IAnyTopic) -> None`

Create a TopicQos with settings equivalent to those of the provided Topic.

3. `__init__(self: rti.connexdds.TopicQos, other: rti.connexdds.TopicQos) -> None`

Create a copy of a TopicQos object.

`__lshift__` (**args*, ***kwargs*)

Overloaded function.

1. `__lshift__(self: rti.connexdds.TopicQos, arg0: rti.connexdds.TopicData) -> rti.connexdds.TopicQos`

Set the TopicDataQoS.

2. `__lshift__(self: rti.connexdds.TopicQos, arg0: rti.connexdds.Durability) -> rti.connexdds.TopicQos`

Set the DurabilityQoS.

3. `__lshift__(self: rti.connexdds.TopicQos, arg0: rti.connexdds.DurabilityService) -> rti.connexdds.TopicQos`

Set the DurabilityServiceQoS.

4. `__lshift__(self: rti.connexdds.TopicQos, arg0: rti.connexdds.Deadline) -> rti.connexdds.TopicQos`

Set the DeadlineQoS.

5. `__lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.LatencyBudget) -> rti.connextdds.TopicQos`

Set the LatencyBudgetQoS.

6. `__lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Liveliness) -> rti.connextdds.TopicQos`

Set the LivelinessQoS.

7. `__lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Reliability) -> rti.connextdds.TopicQos`

Set the ReliabilityQoS.

8. `__lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.DestinationOrder) -> rti.connextdds.TopicQos`

Set the DestinationOrderQoS.

9. `__lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.History) -> rti.connextdds.TopicQos`

Set the HistoryQoS.

10. `__lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.ResourceLimits) -> rti.connextdds.TopicQos`

Set the ResourceLimitsQoS.

11. `__lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.TransportPriority) -> rti.connextdds.TopicQos`

Set the TransportPriorityQoS.

12. `__lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Lifespan) -> rti.connextdds.TopicQos`

Set the LifespanQoS.

13. `__lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Ownership) -> rti.connextdds.TopicQos`

Set the OwnershipQoS.

14. `__lshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.DataRepresentation) -> rti.connextdds.TopicQos`

Set the DataRepresentationQoS.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.TopicQos) -> bool`

Test for inequality.

`__rshift__(*args, **kwargs)`

Overloaded function.

1. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.TopicData) -> rti.connextdds.TopicQos`

Get the TopicDataQoS.

2. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Durability) -> rti.connextdds.TopicQos`

Get the DurabilityQoS.

3. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.DurabilityService) -> rti.connextdds.TopicQos`

Get the DurabilityServiceQoS.

4. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Deadline) -> rti.connextdds.TopicQos`

Get the DeadlineQoS.

5. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.LatencyBudget) -> rti.connextdds.TopicQos`

Get the LatencyBudgetQoS.

6. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Liveliness) -> rti.connextdds.TopicQos`

Get the LivelinessQoS.

7. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Reliability) -> rti.connextdds.TopicQos`

Get the ReliabilityQoS.

8. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.DestinationOrder) -> rti.connextdds.TopicQos`

Get the DestinationOrderQoS.

9. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.History) -> rti.connextdds.TopicQos`

Get the HistoryQoS.

10. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.ResourceLimits) -> rti.connextdds.TopicQos`

Get the ResourceLimitsQoS.

11. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.TransportPriority) -> rti.connextdds.TopicQos`

Get the TransportPriorityQoS.

12. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Lifespan) -> rti.connextdds.TopicQos`

Get the LifespanQoS.

13. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.Ownership) -> rti.connextdds.TopicQos`

Get the OwnershipQoS.

14. `__rshift__(self: rti.connextdds.TopicQos, arg0: rti.connextdds.DataRepresentation) -> rti.connextdds.TopicQos`

Get the DataRepresentationQoS.

`__str__ (self: rti.connextdds.TopicQos) → str`

property data_representation

Get/set DataRepresentation QoS.

property deadline

Get/set Deadline QoS.

property destination_order

Get/set DestinationOrder QoS.

property durability

Get/set Durability QoS.

property durability_service

Get/set DurabilityService QoS.

property history

Get/set History QoS.

property latency_budget

Get/set LatencyBudget QoS.

property lifespan

Get/set Lifespan QoS.

property liveliness

Get/set Liveliness QoS.

property ownership

Get/set Ownership QoS.

property reliability

Get/set Reliability QoS.

property resource_limits

Get/set ResourceLimits QoS.

`to_string (self: rti.connextdds.TopicQos, format: rti.connextdds.QosPrintFormat = QosPrintFormat(), base: Optional[rti.connextdds.TopicQos] = None, print_all: bool = False) → str`

Convert QoS to string based on params.

property topic_data

Get/set TopicData QoS.

property transport_priority

Get/set TransportPriority QoS.

class `rti.connexdds.TopicQuery`

Bases: `pybind11_object`

__enter__ (*self*: `rti.connexdds.TopicQuery`) → `rti.connexdds.TopicQuery`

Enter a context managed block for a TopicQuery.

__eq__ (*self*: `rti.connexdds.TopicQuery`, *arg0*: `rti.connexdds.TopicQuery`) → `bool`

Compare DataStateEx objects for equality.

__exit__ (*self*: `rti.connexdds.TopicQuery`, *arg0*: *object*, *arg1*: *object*, *arg2*: *object*) → `None`

Exit a context managed block for a TopicQuery.

__hash__ = `None`

__init__ (*self*: `rti.connexdds.TopicQuery`, *reader*: `rti.connexdds.IAnyDataReader`, *selection*: `rti.connexdds.TopicQuerySelection`) → `None`

Creates a TopicQuery for a given DataReader.

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.TopicQuery`, *arg0*: `rti.connexdds.TopicQuery`) → `bool`

Compare DataStateEx objects for inequality.

close (*self*: `rti.connexdds.TopicQuery`) → `None`

Deletes and cancels this TopicQuery.

property closed

Indicates whether this TopicQuery has been closed with close().

property datareader

Gets the DataReader associated to this TopicQuery.

static find (*reader*: `rti.connexdds.IAnyDataReader`, *guid*: `rti.connexdds.Guid`) → `Optional[rti.connexdds.TopicQuery]`

Lookup a TopicQuery by its GUID in the reader that created it

property guid

The TopicQuery GUID.

retain (*self*: `rti.connexdds.TopicQuery`) → `None`

Disable automatic destruction of this TopicQuery.

static select_all (*reader*: `rti.connexdds.IAnyDataReader`) → `rti.connexdds.TopicQuery`

Create a TopicQuery that requests all data.

unretain (*self*: `rti.connexdds.TopicQuery`) → `None`

static use_reader_content_filter (*reader*: rti.connextdds.IAnyDataReader) →
rti.connextdds.TopicQuery

Create a TopicQuery with a DataReader's content filter.

class rti.connextdds.**TopicQueryData**

Bases: pybind11_object

__init__ (*args, **kwargs)

__module__ = 'rti.connextdds'

static create_from_service_request (*service_request*: rti.connextdds.ServiceRequest)
 → *rti.connextdds.TopicQueryData*

Creates a TopicQueryData from a ServiceRequest.

property original_related_reader_guid

Identifies the DataReader that created the TopicQuery.

property selection

The data selection.

property topic_name

The topic name of the DataReader.

class rti.connextdds.**TopicQueryDispatch**

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.TopicQueryDispatch, *arg0*: rti.connextdds.TopicQueryDispatch) →
 bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.TopicQueryDispatch) -> None

Creates the default policy.

2. **__init__**(*self*: rti.connextdds.TopicQueryDispatch, *enable*: bool, *publication_period*: rti.connextdds.Duration, *samples_per_period*: int) -> None

Creates a policy with the provided values for *enable*, *publication_period* and *samples_per_period*.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.TopicQueryDispatch, *arg0*: rti.connextdds.TopicQueryDispatch) →
 bool

Test for inequality.

property enable

Allows this writer to dispatch TopicQueries.

property publication_period

The periodic interval at which samples are published.

property samples_per_period

The maximum number of samples to publish in each publication_period.

class rti.connextdds.TopicQuerySelection

Bases: pybind11_object

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.TopicQuerySelection, filter: rti.connextdds.Filter) -> None`

Creates a TopicQuerySelection.

2. `__init__(self: rti.connextdds.TopicQuerySelection, filter: rti.connextdds.Filter, kind: rti.connextdds.TopicQuerySelectionKind) -> None`

Creates a TopicQuerySelection with a selection kind.

__module__ = 'rti.connextdds'

property filter

The filter.

property kind

Indicates whether the sample selection is limited to cached samples or not.

class rti.connextdds.TopicQuerySelectionKind

Bases: pybind11_object

CONTINUOUS = <TopicQuerySelectionKind.CONTINUOUS: 1>

HISTORY_SNAPSHOT = <TopicQuerySelectionKind.HISTORY_SNAPSHOT: 0>

class TopicQuerySelectionKind

Bases: pybind11_object

Members:

HISTORY_SNAPSHOT : [default] Indicates that the TopicQuery may only select samples that were in the DataWriter cache upon reception.

CONTINUOUS : Indicates that the TopicQuery may continue selecting samples published after its reception.

The subscribing application must explicitly delete the TopicQuery (see TopicQuery.close()) to signal the DataWriters to stop publishing samples for it.

CONTINUOUS = <TopicQuerySelectionKind.CONTINUOUS: 1>

HISTORY_SNAPSHOT = <TopicQuerySelectionKind.HISTORY_SNAPSHOT: 0>

```

__eq__ (self: object, other: object) → bool
__getstate__ (self: object) → int
__hash__ (self: object) → int
__index__ (self: rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind) → int
__init__ (self: rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind, value:
          int) → None
__int__ (self: rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind) → int
__members__ = {'CONTINUOUS':
<TopicQuerySelectionKind.CONTINUOUS: 1>, 'HISTORY_SNAPSHOT':
<TopicQuerySelectionKind.HISTORY_SNAPSHOT: 0>}
__module__ = 'rti.connexdds'
__ne__ (self: object, other: object) → bool
__repr__ (self: object) → str
__setstate__ (self: rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind,
              state: int) → None
__str__ ()
          name(self: handle) -> str

```

property name

property value

```

__eq__ (self: rti.connexdds.TopicQuerySelectionKind, arg0:
        rti.connexdds.TopicQuerySelectionKind) → bool

```

Apply operator to underlying enumerated values.

```

__ge__ (self: rti.connexdds.TopicQuerySelectionKind, arg0:
        rti.connexdds.TopicQuerySelectionKind) → bool

```

Apply operator to underlying enumerated values.

```

__gt__ (self: rti.connexdds.TopicQuerySelectionKind, arg0:
        rti.connexdds.TopicQuerySelectionKind) → bool

```

Apply operator to underlying enumerated values.

```

__hash__ = None

```

```

__init__ (*args, **kwargs)

```

Overloaded function.

1. `__init__(self: rti.connexdds.TopicQuerySelectionKind) -> None`

Initializes enum to 0.

2. `__init__(self: rti.connextdds.TopicQuerySelectionKind, arg0: rti.connextdds.TopicQuerySelectionKind) -> None`

Copy constructor.

`__int__(self: rti.connextdds.TopicQuerySelectionKind) -> rti.connextdds.TopicQuerySelectionKind`

Int conversion.

`__le__(self: rti.connextdds.TopicQuerySelectionKind, arg0: rti.connextdds.TopicQuerySelectionKind) -> bool`

Apply operator to underlying enumerated values.

`__lt__(self: rti.connextdds.TopicQuerySelectionKind, arg0: rti.connextdds.TopicQuerySelectionKind) -> bool`

Apply operator to underlying enumerated values.

`__module__ = 'rti.connextdds'`

`__ne__(self: rti.connextdds.TopicQuerySelectionKind, arg0: rti.connextdds.TopicQuerySelectionKind) -> bool`

Apply operator to underlying enumerated values.

`__str__(self: rti.connextdds.TopicQuerySelectionKind) -> str`
String conversion.

property underlying

Retrieves the actual enumerated value.

class `rti.connextdds.TopicSeq`

Bases: `pybind11_object`

`__add__(self: List[rti.connextdds.Topic], arg0: List[rti.connextdds.Topic]) -> List[rti.connextdds.Topic]`

`__bool__(self: List[rti.connextdds.Topic]) -> bool`

Check whether the list is nonempty

`__contains__(self: List[rti.connextdds.Topic], x: rti.connextdds.Topic) -> bool`

Return true the container contains `x`

`__delitem__(*args, **kwargs)`

Overloaded function.

1. `__delitem__(self: List[rti.connextdds.Topic], arg0: int) -> None`

Delete the list elements at index `i`

2. `__delitem__(self: List[rti.connextdds.Topic], arg0: slice) -> None`

Delete list elements using a slice object

`__eq__(self: List[rti.connextdds.Topic], arg0: List[rti.connextdds.Topic]) -> bool`

__getitem__ (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: List[rti.connextdds.Topic], s: slice) -> List[rti.connextdds.Topic]`

Retrieve list elements using a slice object

2. `__getitem__(self: List[rti.connextdds.Topic], arg0: int) -> rti.connextdds.Topic`

__hash__ = None

__iadd__ (self: List[rti.connextdds.Topic], arg0: List[rti.connextdds.Topic]) → List[rti.connextdds.Topic]

__imul__ (self: List[rti.connextdds.Topic], arg0: int) → List[rti.connextdds.Topic]

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.TopicSeq) -> None`
2. `__init__(self: rti.connextdds.TopicSeq, arg0: List[rti.connextdds.Topic]) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.TopicSeq, arg0: Iterable) -> None`

__iter__ (self: List[rti.connextdds.Topic]) → Iterator

__len__ (self: List[rti.connextdds.Topic]) → int

__module__ = 'rti.connextdds'

__mul__ (self: List[rti.connextdds.Topic], arg0: int) → List[rti.connextdds.Topic]

__ne__ (self: List[rti.connextdds.Topic], arg0: List[rti.connextdds.Topic]) → bool

__rmul__ (self: List[rti.connextdds.Topic], arg0: int) → List[rti.connextdds.Topic]

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: List[rti.connextdds.Topic], arg0: int, arg1: rti.connextdds.Topic) -> None`
2. `__setitem__(self: List[rti.connextdds.Topic], arg0: slice, arg1: List[rti.connextdds.Topic]) -> None`

Assign list elements using a slice object

append (self: List[rti.connextdds.Topic], x: rti.connextdds.Topic) → None

Add an item to the end of the list

clear (self: List[rti.connextdds.Topic]) → None

Clear the contents

count (*self*: List[rti.connexdds.Topic], *x*: rti.connexdds.Topic) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: List[rti.connexdds.Topic], *L*: List[rti.connexdds.Topic]) -> None

Extend the list by appending all the items in the given list

2. extend(*self*: List[rti.connexdds.Topic], *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: List[rti.connexdds.Topic], *i*: int, *x*: rti.connexdds.Topic) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: List[rti.connexdds.Topic]) -> rti.connexdds.Topic

Remove and return the last item

2. pop(*self*: List[rti.connexdds.Topic], *i*: int) -> rti.connexdds.Topic

Remove and return the item at index *i*

remove (*self*: List[rti.connexdds.Topic], *x*: rti.connexdds.Topic) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.TransportBuiltin

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.TransportBuiltin, *arg0*: rti.connexdds.TransportBuiltin) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. __init__(*self*: rti.connexdds.TransportBuiltin) -> None

Creates the default policy.

2. __init__(*self*: rti.connexdds.TransportBuiltin, *mask*: rti.connexdds.TransportBuiltinMask) -> None

Creates an instance with the transport selection that the mask specifies.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.TransportBuiltin, *arg0*: rti.connexdds.TransportBuiltin) → bool

Test for inequality.

`all = <rti.connexdds.TransportBuiltin object>`

property mask

The selected transports through a mask.

`none = <rti.connexdds.TransportBuiltin object>`

`shmem = <rti.connexdds.TransportBuiltin object>`

`udpv4 = <rti.connexdds.TransportBuiltin object>`

`udpv6 = <rti.connexdds.TransportBuiltin object>`

class `rti.connexdds.TransportBuiltinMask`

Bases: `pybind11_object`

ALL = `11111111`

NONE = `00000000`

SHMEM = `00000010`

UDPv4 = `00000001`

UDPv6 = `00001000`

__and__ (*self*: `rti.connexdds.TransportBuiltinMask`, *arg0*: `rti.connexdds.TransportBuiltinMask`) → `rti.connexdds.TransportBuiltinMask`

Bitwise logical AND of masks.

__bool__ (*self*: `rti.connexdds.TransportBuiltinMask`) → `bool`

Test if any bits are set.

__contains__ (*self*: `rti.connexdds.TransportBuiltinMask`, *arg0*: `rti.connexdds.TransportBuiltinMask`) → `bool`

__eq__ (*self*: `rti.connexdds.TransportBuiltinMask`, *arg0*: `rti.connexdds.TransportBuiltinMask`) → `bool`

Compare masks for equality.

__getitem__ (*self*: `rti.connexdds.TransportBuiltinMask`, *arg0*: `int`) → `bool`

Get individual mask bit.

__hash__ = `None`

__iand__ (*self*: `rti.connexdds.TransportBuiltinMask`, *arg0*: `rti.connexdds.TransportBuiltinMask`) → `rti.connexdds.TransportBuiltinMask`

Set mask to logical AND with another mask.

__ilshift__ (*self*: `rti.connexdds.TransportBuiltinMask`, *arg0*: `int`) → `rti.connexdds.TransportBuiltinMask`

Left shift bits in mask.

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.TransportBuiltinMask) -> None

Create a TransportBuiltinMask equivalent to TransportBuiltinMask.NONE

2. **__init__**(self: rti.connexdds.TransportBuiltinMask, value: int) -> None

Creates a mask from the bits in an integer.

__int__ (self: rti.connexdds.TransportBuiltinMask) → int

Convert mask to int.

__ior__ (self: rti.connexdds.TransportBuiltinMask, arg0: rti.connexdds.TransportBuiltinMask) →
rti.connexdds.TransportBuiltinMask

Set mask to logical OR with another mask.

__irshift__ (self: rti.connexdds.TransportBuiltinMask, arg0: int) →
rti.connexdds.TransportBuiltinMask

Right shift bits in mask.

__ixor__ (self: rti.connexdds.TransportBuiltinMask, arg0: rti.connexdds.TransportBuiltinMask)
→ *rti.connexdds.TransportBuiltinMask*

Set mask to logical XOR with another mask.

__lshift__ (self: rti.connexdds.TransportBuiltinMask, arg0: int) →
rti.connexdds.TransportBuiltinMask

Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.TransportBuiltinMask, arg0: rti.connexdds.TransportBuiltinMask) →
bool

Compare masks for inequality.

__or__ (self: rti.connexdds.TransportBuiltinMask, arg0: rti.connexdds.TransportBuiltinMask) →
rti.connexdds.TransportBuiltinMask

Bitwise logical OR of masks.

__repr__ (self: rti.connexdds.TransportBuiltinMask) → str

Convert mask to string.

__rshift__ (self: rti.connexdds.TransportBuiltinMask, arg0: int) →
rti.connexdds.TransportBuiltinMask

Right shift bits in mask.

__setitem__ (self: rti.connexdds.TransportBuiltinMask, arg0: int, arg1: bool) → None

Set individual mask bit

__str__ (self: rti.connexdds.TransportBuiltinMask) → str

Convert mask to string.

__xor__ (*self*: rti.connexdds.TransportBuiltinMask, *arg0*: rti.connexdds.TransportBuiltinMask) → *rti.connexdds.TransportBuiltinMask*

Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (**args*, ***kwargs*)

Overloaded function.

1. flip(*self*: rti.connexdds.TransportBuiltinMask) -> rti.connexdds.TransportBuiltinMask

Flip all bits in the mask.

2. flip(*self*: rti.connexdds.TransportBuiltinMask, *pos*: int) -> rti.connexdds.TransportBuiltinMask

Flip the mask bit at the specified position.

reset (**args*, ***kwargs*)

Overloaded function.

1. reset(*self*: rti.connexdds.TransportBuiltinMask) -> rti.connexdds.TransportBuiltinMask

Clear all bits in the mask.

2. reset(*self*: rti.connexdds.TransportBuiltinMask, *pos*: int) -> rti.connexdds.TransportBuiltinMask

Clear the mask bit at the specified position.

set (**args*, ***kwargs*)

Overloaded function.

1. set(*self*: rti.connexdds.TransportBuiltinMask) -> rti.connexdds.TransportBuiltinMask

Set all bits in the mask.

2. set(*self*: rti.connexdds.TransportBuiltinMask, *pos*: int, *value*: bool = True) -> rti.connexdds.TransportBuiltinMask

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (*self*: rti.connexdds.TransportBuiltinMask, *pos*: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (*self*: rti.connexdds.TransportBuiltinMask) → bool

Test if all bits are set.

test_any (*self*: rti.connexdds.TransportBuiltinMask) → bool

Test if any bits are set.

test_none (*self*: rti.connextdds.TransportBuiltinMask) → bool
 Test if none of the bits are set.

```
class rti.connextdds.TransportClassId
    Bases: pybind11_object
    ANY = <TransportClassId.ANY: 0>
    INTRA = <TransportClassId.INTRA: 3>
    INVALID = <TransportClassId.INVALID: -1>
    RESERVED_RANGE = <TransportClassId.RESERVED_RANGE: 1000>
    SHMEM = <TransportClassId.SHMEM: 16777216>
    SHMEM_510 = <TransportClassId.SHMEM_510: 2>
    TCPV4_LAN = <TransportClassId.TCPV4_LAN: 8>
    TCPV4_WAN = <TransportClassId.TCPV4_WAN: 9>
    TLSV4_LAN = <TransportClassId.TLSV4_LAN: 10>
    TLSV4_WAN = <TransportClassId.TLSV4_WAN: 11>
```

```
class TransportClassId
    Bases: pybind11_object
    Members:
    INVALID : Invalid.
    ANY : Any.
    UDPv4 : UDPv4.
    UDPv4_WAN : UDPv4 WAN.
    SHMEM : Shared memory.
    SHMEM_510 : Shared Memory from 5.1.0 and earlier.
    INTRA : Intra-participant.
    UDPv6 : UDPv6.
    UDPv6_510 : UDPv6 for 5.1.0 and earlier.
    TCPV4_LAN : TCPv4 LAN mode.
    TCPV4_WAN : TCPv4 WAN mode.
    TLSV4_LAN : TCPv4 LAN mode with TLS.
    TLSV4_WAN : TCPv4 WAN mode with TLS.
    RESERVED_RANGE : eserved for additional user-defined transport plugins.
```

```

ANY = <TransportClassId.ANY: 0>
INTRA = <TransportClassId.INTRA: 3>
INVALID = <TransportClassId.INVALID: -1>
RESERVED_RANGE = <TransportClassId.RESERVED_RANGE: 1000>
SHMEM = <TransportClassId.SHMEM: 16777216>
SHMEM_510 = <TransportClassId.SHMEM_510: 2>
TCPV4_LAN = <TransportClassId.TCPV4_LAN: 8>
TCPV4_WAN = <TransportClassId.TCPV4_WAN: 9>
TLSV4_LAN = <TransportClassId.TLSV4_LAN: 10>
TLSV4_WAN = <TransportClassId.TLSV4_WAN: 11>
UDPv4 = <TransportClassId.UDPv4: 1>
UDPv4_WAN = <TransportClassId.UDPv4_WAN: 16777217>
UDPv6 = <TransportClassId.SHMEM_510: 2>
UDPv6_510 = <TransportClassId.UDPv6_510: 5>

__eq__(self: object, other: object) → bool
__getstate__(self: object) → int
__hash__(self: object) → int
__index__(self: rti.connexdds.TransportClassId.TransportClassId) → int
__init__(self: rti.connexdds.TransportClassId.TransportClassId, value: int) → None
__int__(self: rti.connexdds.TransportClassId.TransportClassId) → int

__members__ = {'ANY': <TransportClassId.ANY: 0>, 'INTRA':
<TransportClassId.INTRA: 3>, 'INVALID':
<TransportClassId.INVALID: -1>, 'RESERVED_RANGE':
<TransportClassId.RESERVED_RANGE: 1000>, 'SHMEM':
<TransportClassId.SHMEM: 16777216>, 'SHMEM_510':
<TransportClassId.SHMEM_510: 2>, 'TCPV4_LAN':
<TransportClassId.TCPV4_LAN: 8>, 'TCPV4_WAN':
<TransportClassId.TCPV4_WAN: 9>, 'TLSV4_LAN':
<TransportClassId.TLSV4_LAN: 10>, 'TLSV4_WAN':
<TransportClassId.TLSV4_WAN: 11>, 'UDPv4':
<TransportClassId.UDPv4: 1>, 'UDPv4_WAN':
<TransportClassId.UDPv4_WAN: 16777217>, 'UDPv6':
<TransportClassId.SHMEM_510: 2>, 'UDPv6_510':
<TransportClassId.UDPv6_510: 5>}

```

```

__module__ = 'rti.connextdds'

__ne__ (self: object, other: object) → bool

__repr__ (self: object) → str

__setstate__ (self: rti.connextdds.TransportClassId.TransportClassId, state: int) → None

__str__ ()
    name(self: handle) -> str

property name

property value

UDPv4 = <TransportClassId.UDPv4: 1>
UDPv4_WAN = <TransportClassId.UDPv4_WAN: 16777217>
UDPv6 = <TransportClassId.SHMEM_510: 2>
UDPv6_510 = <TransportClassId.UDPv6_510: 5>

__eq__ (self: rti.connextdds.TransportClassId, arg0: rti.connextdds.TransportClassId) → bool
    Apply operator to underlying enumerated values.

__ge__ (self: rti.connextdds.TransportClassId, arg0: rti.connextdds.TransportClassId) → bool
    Apply operator to underlying enumerated values.

__gt__ (self: rti.connextdds.TransportClassId, arg0: rti.connextdds.TransportClassId) → bool
    Apply operator to underlying enumerated values.

__hash__ = None

__init__ (*args, **kwargs)
    Overloaded function.
    1. __init__(self: rti.connextdds.TransportClassId) -> None
        Initializes enum to 0.
    2. __init__(self: rti.connextdds.TransportClassId, arg0: rti.connextdds.TransportClassId.TransportClassId) -> None
        Copy constructor.

__int__ (self: rti.connextdds.TransportClassId) →
    rti.connextdds.TransportClassId.TransportClassId
    Int conversion.

__le__ (self: rti.connextdds.TransportClassId, arg0: rti.connextdds.TransportClassId) → bool
    Apply operator to underlying enumerated values.

```

__lt__ (*self*: rti.connexdds.TransportClassId, *arg0*: rti.connexdds.TransportClassId) → bool
Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.TransportClassId, *arg0*: rti.connexdds.TransportClassId) → bool
Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.TransportClassId) → str
String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.TransportInfo

Bases: pybind11_object

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.TransportInfo) -> None

Creates an instance with default values.

2. **__init__**(*self*: rti.connexdds.TransportInfo, *class_id*: rti.connexdds.TransportClassId.TransportClassId, *message_size_max*: int) -> None

Creates an instance with the specified class id and max size.

__module__ = 'rti.connexdds'

property class_id

The *class_id* identifies the transport associated with the *message_size_max*.

property message_size_max

The maximum size of an RTPS message in bytes that can be sent or received by the transport plugin identified by the *class_id*.

class rti.connexdds.TransportInfoSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.TransportInfoSeq, *arg0*: rti.connexdds.TransportInfoSeq) → *rti.connexdds.TransportInfoSeq*

__bool__ (*self*: rti.connexdds.TransportInfoSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.TransportInfoSeq, *x*: rti.connexdds.TransportInfo) → bool

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.TransportInfoSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. `__delitem__(self: rti.connexdds.TransportInfoSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__(self: rti.connexdds.TransportInfoSeq, arg0: rti.connexdds.TransportInfoSeq) → bool`

`__getitem__(*args, **kwargs)`

Overloaded function.

1. `__getitem__(self: rti.connexdds.TransportInfoSeq, s: slice) -> rti.connexdds.TransportInfoSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.TransportInfoSeq, arg0: int) -> rti.connexdds.TransportInfo`

`__hash__ = None`

`__iadd__(self: rti.connexdds.TransportInfoSeq, arg0: rti.connexdds.TransportInfoSeq) → rti.connexdds.TransportInfoSeq`

`__imul__(self: rti.connexdds.TransportInfoSeq, arg0: int) → rti.connexdds.TransportInfoSeq`

`__init__(*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connexdds.TransportInfoSeq) -> None`
2. `__init__(self: rti.connexdds.TransportInfoSeq, arg0: rti.connexdds.TransportInfoSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.TransportInfoSeq, arg0: Iterable) -> None`

`__iter__(self: rti.connexdds.TransportInfoSeq) → Iterator`

`__len__(self: rti.connexdds.TransportInfoSeq) → int`

`__module__ = 'rti.connexdds'`

`__mul__(self: rti.connexdds.TransportInfoSeq, arg0: int) → rti.connexdds.TransportInfoSeq`

`__ne__(self: rti.connexdds.TransportInfoSeq, arg0: rti.connexdds.TransportInfoSeq) → bool`

`__rmul__(self: rti.connexdds.TransportInfoSeq, arg0: int) → rti.connexdds.TransportInfoSeq`

`__setitem__(*args, **kwargs)`

Overloaded function.

1. `__setitem__(self: rti.connexdds.TransportInfoSeq, arg0: int, arg1: rti.connexdds.TransportInfo) -> None`

2. `__setitem__(self: rti.connexdds.TransportInfoSeq, arg0: slice, arg1: rti.connexdds.TransportInfoSeq) -> None`

Assign list elements using a slice object

append (*self*: rti.connexdds.TransportInfoSeq, *x*: rti.connexdds.TransportInfo) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.TransportInfoSeq) → None

Clear the contents

count (*self*: rti.connexdds.TransportInfoSeq, *x*: rti.connexdds.TransportInfo) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. `extend(self: rti.connexdds.TransportInfoSeq, L: rti.connexdds.TransportInfoSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connexdds.TransportInfoSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.TransportInfoSeq, *i*: int, *x*: rti.connexdds.TransportInfo) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connexdds.TransportInfoSeq) -> rti.connexdds.TransportInfo`

Remove and return the last item

2. `pop(self: rti.connexdds.TransportInfoSeq, i: int) -> rti.connexdds.TransportInfo`

Remove and return the item at index *i*

remove (*self*: rti.connexdds.TransportInfoSeq, *x*: rti.connexdds.TransportInfo) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class `rti.connexdds.TransportInfoVector`

Bases: `pybind11_object`

A DDS standard container with functionality similar to a C++ vector.

__eq__ (*self*: rti.connexdds.TransportInfoVector, *arg0*: rti.connexdds.TransportInfoVector) → bool

Compare TransportInfoVectors for equality.

__getitem__ (*self*: rti.connexdds.TransportInfoVector, *arg0*: int) → *rti.connexdds.TransportInfo*

Get the value at the specified index.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.TransportInfoVector) -> None

Create an empty TransportInfoVector

2. **__init__**(self: rti.connexdds.TransportInfoVector, size: int) -> None

Create a TransportInfoVector with a preallocated size.

3. **__init__**(self: rti.connexdds.TransportInfoVector, vector: rti.connexdds.TransportInfoVector) -> None

Create a copy from another TransportInfoVector.

4. **__init__**(self: rti.connexdds.TransportInfoVector, arg0: Iterable) -> None

5. **__init__**(self: rti.connexdds.TransportInfoVector, list: rti.connexdds.TransportInfoSeq) -> None

Create a TransportInfoVector from a list of values.

__iter__ (self: rti.connexdds.TransportInfoVector) → Iterator

Iterate over the contents of the vector.

__len__ (self: rti.connexdds.TransportInfoVector) → int

Get the length of the TransportInfoVector.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.TransportInfoVector, arg0: rti.connexdds.TransportInfoVector) → bool

Compare TransportInfoVectors for inequality.

__setitem__ (self: rti.connexdds.TransportInfoVector, arg0: int, arg1: rti.connexdds.TransportInfo) → None

Set the value at the specified index.

clear (self: rti.connexdds.TransportInfoVector) → None

Resize TransportInfoVector to 0.

resize (self: rti.connexdds.TransportInfoVector, size: int) → None

Resize TransportInfoVector.

class rti.connexdds.TransportMulticast

Bases: pybind11_object

__eq__ (self: rti.connexdds.TransportMulticast, arg0: rti.connexdds.TransportMulticast) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.TransportMulticast) -> None

Creates the default policy.

2. `__init__(self: rti.connextdds.TransportMulticast, value: rti.connextdds.TransportMulticastSettingsSeq, kind: rti.connextdds.TransportMulticastKind) -> None`

Creates an instance with the specified multicast settings.

```
__module__ = 'rti.connextdds'
```

```
__ne__ (self: rti.connextdds.TransportMulticast, arg0: rti.connextdds.TransportMulticast) → bool
    Test for inequality.
```

property kind

A value that specifies a way to determine how to obtain the multicast address.

property value

A sequence of multicast communications settings.

```
class rti.connextdds.TransportMulticastKind
```

```
Bases: pybind11_object
```

```
AUTOMATIC = <TransportMulticastKind.AUTOMATIC: 0>
```

```
class TransportMulticastKind
```

```
Bases: pybind11_object
```

Members:

AUTOMATIC : Selects the multicast address automatically.

NOTE: This setting is required when using the TransportMulticastMapping.

UNICAST : Selects a unicast-only mode.

```
AUTOMATIC = <TransportMulticastKind.AUTOMATIC: 0>
```

```
UNICAST = <TransportMulticastKind.UNICAST: 1>
```

```
__eq__ (self: object, other: object) → bool
```

```
__getstate__ (self: object) → int
```

```
__hash__ (self: object) → int
```

```
__index__ (self: rti.connextdds.TransportMulticastKind.TransportMulticastKind) → int
```

```
__init__ (self: rti.connextdds.TransportMulticastKind.TransportMulticastKind, value: int)
    → None
```

```
__int__ (self: rti.connextdds.TransportMulticastKind.TransportMulticastKind) → int
```

```
__members__ = {'AUTOMATIC': <TransportMulticastKind.AUTOMATIC: 0>, 'UNICAST': <TransportMulticastKind.UNICAST: 1>}
```

```
__module__ = 'rti.connextdds'
```

`__ne__` (*self: object, other: object*) → bool

`__repr__` (*self: object*) → str

`__setstate__` (*self: rti.connextdds.TransportMulticastKind.TransportMulticastKind, state: int*) → None

`__str__` ()

name(self: handle) -> str

property name

property value

UNICAST = <TransportMulticastKind.UNICAST: 1>

`__eq__` (*self: rti.connextdds.TransportMulticastKind, arg0: rti.connextdds.TransportMulticastKind*) → bool

Apply operator to underlying enumerated values.

`__ge__` (*self: rti.connextdds.TransportMulticastKind, arg0: rti.connextdds.TransportMulticastKind*) → bool

Apply operator to underlying enumerated values.

`__gt__` (*self: rti.connextdds.TransportMulticastKind, arg0: rti.connextdds.TransportMulticastKind*) → bool

Apply operator to underlying enumerated values.

`__hash__` = None

`__init__` (**args, **kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.TransportMulticastKind) -> None`

Initializes enum to 0.

2. `__init__(self: rti.connextdds.TransportMulticastKind, arg0: rti.connextdds.TransportMulticastKind.TransportMulticastKind) -> None`

Copy constructor.

`__int__` (*self: rti.connextdds.TransportMulticastKind*) → *rti.connextdds.TransportMulticastKind.TransportMulticastKind*

Int conversion.

`__le__` (*self: rti.connextdds.TransportMulticastKind, arg0: rti.connextdds.TransportMulticastKind*) → bool

Apply operator to underlying enumerated values.

`__lt__` (*self: rti.connextdds.TransportMulticastKind, arg0: rti.connextdds.TransportMulticastKind*) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.TransportMulticastKind, *arg0*: rti.connexdds.TransportMulticastKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.TransportMulticastKind) → str
String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.**TransportMulticastMapping**

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.TransportMulticastMapping, *arg0*: rti.connexdds.TransportMulticastMapping) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.TransportMulticastMapping) -> None

Creates the default policy.

2. **__init__**(*self*: rti.connexdds.TransportMulticastMapping, *mappings*: rti.connexdds.MulticastMappingSeq) -> None

Creates an object with with specified mappings.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.TransportMulticastMapping, *arg0*: rti.connexdds.TransportMulticastMapping) → bool

Test for inequality.

property value

A sequence of transport multicast mappings.

class rti.connexdds.**TransportMulticastMappingFunction**

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.TransportMulticastMappingFunction, *arg0*: rti.connexdds.TransportMulticastMappingFunction) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.TransportMulticastMappingFunction, *dll*: str, *function_name*: str) → None

Create a mapping function instance.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.TransportMulticastMappingFunction, *arg0*: rti.connexdds.TransportMulticastMappingFunction) → bool

Test for inequality.

property dll

The name of the dynamic library containing the mapping function.

property function_name

The name of the mapping function.

class rti.connexdds.TransportMulticastSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.TransportMulticastSeq, *arg0*: rti.connexdds.TransportMulticastSeq) → rti.connexdds.TransportMulticastSeq

__bool__ (*self*: rti.connexdds.TransportMulticastSeq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.TransportMulticastSeq, *x*: rti.connexdds.TransportMulticast) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.TransportMulticastSeq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.TransportMulticastSeq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.TransportMulticastSeq, *arg0*: rti.connexdds.TransportMulticastSeq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.TransportMulticastSeq, *s*: slice) -> rti.connexdds.TransportMulticastSeq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.TransportMulticastSeq, *arg0*: int) -> rti.connexdds.TransportMulticast

__hash__ = None

__iadd__ (*self*: rti.connexdds.TransportMulticastSeq, *arg0*: rti.connexdds.TransportMulticastSeq) → *rti.connexdds.TransportMulticastSeq*

__imul__ (*self*: rti.connexdds.TransportMulticastSeq, *arg0*: int) → *rti.connexdds.TransportMulticastSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.TransportMulticastSeq) -> None
2. **__init__**(*self*: rti.connexdds.TransportMulticastSeq, *arg0*: rti.connexdds.TransportMulticastSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.TransportMulticastSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.TransportMulticastSeq) → Iterator

__len__ (*self*: rti.connexdds.TransportMulticastSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.TransportMulticastSeq, *arg0*: int) → *rti.connexdds.TransportMulticastSeq*

__ne__ (*self*: rti.connexdds.TransportMulticastSeq, *arg0*: rti.connexdds.TransportMulticastSeq) → bool

__rmul__ (*self*: rti.connexdds.TransportMulticastSeq, *arg0*: int) → *rti.connexdds.TransportMulticastSeq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.TransportMulticastSeq, *arg0*: int, *arg1*: rti.connexdds.TransportMulticast) -> None
2. **__setitem__**(*self*: rti.connexdds.TransportMulticastSeq, *arg0*: slice, *arg1*: rti.connexdds.TransportMulticastSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.TransportMulticastSeq, *x*: rti.connexdds.TransportMulticast) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.TransportMulticastSeq) → None

Clear the contents

count (*self*: rti.connexdds.TransportMulticastSeq, *x*: rti.connexdds.TransportMulticast) → int

Return the number of times *x* appears in the list

extend (*args, **kwargs)

Overloaded function.

1. extend(self: rti.connexdds.TransportMulticastSeq, L: rti.connexdds.TransportMulticastSeq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connexdds.TransportMulticastSeq, L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (self: rti.connexdds.TransportMulticastSeq, i: int, x: rti.connexdds.TransportMulticast) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connexdds.TransportMulticastSeq) -> rti.connexdds.TransportMulticast

Remove and return the last item

2. pop(self: rti.connexdds.TransportMulticastSeq, i: int) -> rti.connexdds.TransportMulticast

Remove and return the item at index *i*

remove (self: rti.connexdds.TransportMulticastSeq, x: rti.connexdds.TransportMulticast) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.TransportMulticastSettings

Bases: pybind11_object

__eq__ (self: rti.connexdds.TransportMulticastSettings, arg0: rti.connexdds.TransportMulticastSettings) → bool

Test for equality.

__hash__ = None

__init__ (self: rti.connexdds.TransportMulticastSettings, transports: rti.connexdds.StringSeq, receive_address: str, receive_port: int) → None

Creates an instance with the specified transport aliases, receive address and receive port.

__module__ = 'rti.connexdds'

__ne__ (self: rti.connexdds.TransportMulticastSettings, arg0: rti.connexdds.TransportMulticastSettings) → bool

Test for inequality.

property receive_address

The multicast group address on which the entity can receive data.

property receive_port

The multicast port on which the entity can receive data.

property transports

A sequence of transport aliases that specifies the transports on which to receive multicast traffic for the entity.

This property's getter returns a deep copy.

class `rti.connexdds.TransportMulticastSettingsSeq`

Bases: `pybind11_object`

__add__ (*self*: `rti.connexdds.TransportMulticastSettingsSeq`, *arg0*: `rti.connexdds.TransportMulticastSettingsSeq`) → `rti.connexdds.TransportMulticastSettingsSeq`

__bool__ (*self*: `rti.connexdds.TransportMulticastSettingsSeq`) → `bool`
Check whether the list is nonempty

__contains__ (*self*: `rti.connexdds.TransportMulticastSettingsSeq`, *x*: `rti.connexdds.TransportMulticastSettings`) → `bool`

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. `__delitem__(self: rti.connexdds.TransportMulticastSettingsSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connexdds.TransportMulticastSettingsSeq, arg0: slice) -> None`

Delete list elements using a slice object

__eq__ (*self*: `rti.connexdds.TransportMulticastSettingsSeq`, *arg0*: `rti.connexdds.TransportMulticastSettingsSeq`) → `bool`

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__(self: rti.connexdds.TransportMulticastSettingsSeq, s: slice) -> rti.connexdds.TransportMulticastSettingsSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.TransportMulticastSettingsSeq, arg0: int) -> rti.connexdds.TransportMulticastSettings`

__hash__ = `None`

__iadd__ (*self*: `rti.connexdds.TransportMulticastSettingsSeq`, *arg0*: `rti.connexdds.TransportMulticastSettingsSeq`) → `rti.connexdds.TransportMulticastSettingsSeq`

__imul__ (*self*: `rti.connexdds.TransportMulticastSettingsSeq`, *arg0*: `int`) → `rti.connexdds.TransportMulticastSettingsSeq`

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(self: rti.connexdds.TransportMulticastSettingsSeq) -> None
2. **__init__**(self: rti.connexdds.TransportMulticastSettingsSeq, arg0: rti.connexdds.TransportMulticastSettingsSeq) -> None

Copy constructor

3. **__init__**(self: rti.connexdds.TransportMulticastSettingsSeq, arg0: Iterable) -> None

__iter__ (self: rti.connexdds.TransportMulticastSettingsSeq) → Iterator

__len__ (self: rti.connexdds.TransportMulticastSettingsSeq) → int

__module__ = 'rti.connexdds'

__mul__ (self: rti.connexdds.TransportMulticastSettingsSeq, arg0: int) →
rti.connexdds.TransportMulticastSettingsSeq

__ne__ (self: rti.connexdds.TransportMulticastSettingsSeq, arg0:
rti.connexdds.TransportMulticastSettingsSeq) → bool

__rmul__ (self: rti.connexdds.TransportMulticastSettingsSeq, arg0: int) →
rti.connexdds.TransportMulticastSettingsSeq

__setitem__ (*args, **kwargs)

Overloaded function.

1. **__setitem__**(self: rti.connexdds.TransportMulticastSettingsSeq, arg0: int, arg1: rti.connexdds.TransportMulticastSettings) -> None
2. **__setitem__**(self: rti.connexdds.TransportMulticastSettingsSeq, arg0: slice, arg1: rti.connexdds.TransportMulticastSettings) -> None

Assign list elements using a slice object

append (self: rti.connexdds.TransportMulticastSettingsSeq, x:
rti.connexdds.TransportMulticastSettings) → None

Add an item to the end of the list

clear (self: rti.connexdds.TransportMulticastSettingsSeq) → None

Clear the contents

count (self: rti.connexdds.TransportMulticastSettingsSeq, x:
rti.connexdds.TransportMulticastSettings) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. **extend**(self: rti.connexdds.TransportMulticastSettingsSeq, L: rti.connexdds.TransportMulticastSettingsSeq) -> None

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.TransportMulticastSettingsSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.TransportMulticastSettingsSeq, *i*: int, *x*: rti.connextdds.TransportMulticastSettings) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. `pop(self: rti.connextdds.TransportMulticastSettingsSeq) -> rti.connextdds.TransportMulticastSettings`

Remove and return the last item

2. `pop(self: rti.connextdds.TransportMulticastSettingsSeq, i: int) -> rti.connextdds.TransportMulticastSettings`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.TransportMulticastSettingsSeq, *x*: rti.connextdds.TransportMulticastSettings) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connextdds.TransportPriority

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.TransportPriority, *arg0*: rti.connextdds.TransportPriority) → bool
Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.TransportPriority) -> None`

Creates a policy with priority 0.

2. `__init__(self: rti.connextdds.TransportPriority, priority: int) -> None`

Creates a policy with the specified priority.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.TransportPriority, *arg0*: rti.connextdds.TransportPriority) → bool
Test for inequality.

property value

The priority.

class `rti.connexdds.TransportSelection`Bases: `pybind11_object``__eq__` (*self*: `rti.connexdds.TransportSelection`, *arg0*: `rti.connexdds.TransportSelection`) → bool
Test for equality.`__hash__` = None`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.TransportSelection) -> None`

Creates the default policy.

2. `__init__(self: rti.connexdds.TransportSelection, enabled_transports: rti.connexdds.StringSeq) -> None`

Creates an instance with the specified transport aliases.

`__module__` = `'rti.connexdds'``__ne__` (*self*: `rti.connexdds.TransportSelection`, *arg0*: `rti.connexdds.TransportSelection`) → bool
Test for inequality.**property value**

A sequence of transport aliases that specifies the transport instances available for use by the entity.

This property's getter returns a deep copy.

class `rti.connexdds.TransportUnicast`Bases: `pybind11_object``__eq__` (*self*: `rti.connexdds.TransportUnicast`, *arg0*: `rti.connexdds.TransportUnicast`) → bool
Test for equality.`__hash__` = None`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.TransportUnicast) -> None`

Creates the default policy.

2. `__init__(self: rti.connexdds.TransportUnicast, settings: rti.connexdds.TransportUnicastSettingsSeq) -> None`

Creates an instance with the specified settings.

`__module__` = `'rti.connexdds'``__ne__` (*self*: `rti.connexdds.TransportUnicast`, *arg0*: `rti.connexdds.TransportUnicast`) → bool
Test for inequality.

property value

The unicast settings.

This property's getter returns a deep copy.

class `rti.connexdds.TransportUnicastSettings`

Bases: `pybind11_object`

__eq__ (*self*: `rti.connexdds.TransportUnicastSettings`, *arg0*: `rti.connexdds.TransportUnicastSettings`) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connexdds.TransportUnicastSettings) -> None`

Creates the default policy.

2. `__init__(self: rti.connexdds.TransportUnicastSettings, transports: rti.connexdds.StringSeq, receive_port: int = 0) -> None`

Creates the default policy.

__module__ = `'rti.connexdds'`

__ne__ (*self*: `rti.connexdds.TransportUnicastSettings`, *arg0*: `rti.connexdds.TransportUnicastSettings`) → bool

Test for inequality.

property receive_port

The unicast port on which the entity can receive data.

property transports

A sequence of transport aliases that specifies the unicast interfaces on which to receive unicast traffic for the entity.

This property's getter returns a deep copy.

class `rti.connexdds.TransportUnicastSettingsSeq`

Bases: `pybind11_object`

__add__ (*self*: `rti.connexdds.TransportUnicastSettingsSeq`, *arg0*: `rti.connexdds.TransportUnicastSettingsSeq`) → `rti.connexdds.TransportUnicastSettingsSeq`

__bool__ (*self*: `rti.connexdds.TransportUnicastSettingsSeq`) → bool

Check whether the list is nonempty

__contains__ (*self*: `rti.connexdds.TransportUnicastSettingsSeq`, *x*: `rti.connexdds.TransportUnicastSettings`) → bool

Return true the container contains x

`__delitem__` (*args, **kwargs)

Overloaded function.

1. `__delitem__(self: rti.connexdds.TransportUnicastSettingsSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connexdds.TransportUnicastSettingsSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__` (self: rti.connexdds.TransportUnicastSettingsSeq, arg0: rti.connexdds.TransportUnicastSettingsSeq) → bool

`__getitem__` (*args, **kwargs)

Overloaded function.

1. `__getitem__(self: rti.connexdds.TransportUnicastSettingsSeq, s: slice) -> rti.connexdds.TransportUnicastSettingsSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connexdds.TransportUnicastSettingsSeq, arg0: int) -> rti.connexdds.TransportUnicastSettings`

`__hash__` = None

`__iadd__` (self: rti.connexdds.TransportUnicastSettingsSeq, arg0: rti.connexdds.TransportUnicastSettingsSeq) → rti.connexdds.TransportUnicastSettingsSeq

`__imul__` (self: rti.connexdds.TransportUnicastSettingsSeq, arg0: int) → rti.connexdds.TransportUnicastSettingsSeq

`__init__` (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connexdds.TransportUnicastSettingsSeq) -> None`
2. `__init__(self: rti.connexdds.TransportUnicastSettingsSeq, arg0: rti.connexdds.TransportUnicastSettingsSeq) -> None`

Copy constructor

3. `__init__(self: rti.connexdds.TransportUnicastSettingsSeq, arg0: Iterable) -> None`

`__iter__` (self: rti.connexdds.TransportUnicastSettingsSeq) → Iterator

`__len__` (self: rti.connexdds.TransportUnicastSettingsSeq) → int

`__module__` = 'rti.connexdds'

`__mul__` (self: rti.connexdds.TransportUnicastSettingsSeq, arg0: int) → rti.connexdds.TransportUnicastSettingsSeq

__ne__ (*self*: rti.connexdds.TransportUnicastSettingsSeq, *arg0*: rti.connexdds.TransportUnicastSettingsSeq) → bool

__rmul__ (*self*: rti.connexdds.TransportUnicastSettingsSeq, *arg0*: int) → rti.connexdds.TransportUnicastSettingsSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.TransportUnicastSettingsSeq, *arg0*: int, *arg1*: rti.connexdds.TransportUnicastSettings) -> None
2. **__setitem__**(*self*: rti.connexdds.TransportUnicastSettingsSeq, *arg0*: slice, *arg1*: rti.connexdds.TransportUnicastSettingsSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.TransportUnicastSettingsSeq, *x*: rti.connexdds.TransportUnicastSettings) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.TransportUnicastSettingsSeq) → None

Clear the contents

count (*self*: rti.connexdds.TransportUnicastSettingsSeq, *x*: rti.connexdds.TransportUnicastSettings) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.TransportUnicastSettingsSeq, *L*: rti.connexdds.TransportUnicastSettingsSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.TransportUnicastSettingsSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.TransportUnicastSettingsSeq, *i*: int, *x*: rti.connexdds.TransportUnicastSettings) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.TransportUnicastSettingsSeq) -> rti.connexdds.TransportUnicastSettings

Remove and return the last item

2. **pop**(*self*: rti.connexdds.TransportUnicastSettingsSeq, *i*: int) -> rti.connexdds.TransportUnicastSettings

Remove and return the item at index *i*

remove (*self*: rti.connexdds.TransportUnicastSettingsSeq, *x*:
rti.connexdds.TransportUnicastSettings) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connexdds.TriggeredConditions

Bases: pybind11_object

__contains__ (*self*: rti.connexdds.TriggeredConditions, *arg0*: rti.connexdds.ICondition) → bool

__getitem__ (*self*: rti.connexdds.TriggeredConditions, *arg0*: int) → rti.connexdds.Condition

__init__ (*args, **kwargs)

__iter__ (*self*: rti.connexdds.TriggeredConditions) → rti.connexdds.TriggeredConditionsIterator

__len__ (*self*: rti.connexdds.TriggeredConditions) → int

__module__ = 'rti.connexdds'

__reversed__ (*self*: rti.connexdds.TriggeredConditions) →
rti.connexdds.TriggeredConditionsIterator

class rti.connexdds.TriggeredConditionsIterator

Bases: pybind11_object

__init__ (*args, **kwargs)

__module__ = 'rti.connexdds'

__next__ (*self*: rti.connexdds.TriggeredConditionsIterator) → rti.connexdds.Condition

Get next triggered condition.

class rti.connexdds.TypeConsistencyEnforcement

Bases: pybind11_object

__eq__ (*self*: rti.connexdds.TypeConsistencyEnforcement, *arg0*:
rti.connexdds.TypeConsistencyEnforcement) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.TypeConsistencyEnforcement) -> None

Creates the default policy.

2. **__init__**(*self*: rti.connexdds.TypeConsistencyEnforcement, *kind*: rti.connexdds.TypeConsistencyEnforcementKind) -> None

Creates an instance with a specific enforcement kind.

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: rti.connexdds.TypeConsistencyEnforcement, arg0:
        rti.connexdds.TypeConsistencyEnforcement) → bool
```

Test for inequality.

```
allow_type_coercion = <rti.connexdds.TypeConsistencyEnforcement
object>
```

```
auto_type_coercion = <rti.connexdds.TypeConsistencyEnforcement
object>
```

```
disallow_type_coercion = <rti.connexdds.TypeConsistencyEnforcement
object>
```

```
property force_type_validation
```

Controls whether type validation is forced.

```
property ignore_enum_literal_names
```

Controls whether enumeration literal names are ignored.

```
property ignore_member_names
```

Controls whether member names are ignored.

```
property ignore_sequence_bounds
```

Controls whether sequence bounds are ignored.

```
property ignore_string_bounds
```

Controls whether string bounds are ignored.

```
property kind
```

The enforcement kind.

```
property prevent_type_widening
```

Controls whether type widening is prevented.

```
class rti.connexdds.TypeConsistencyEnforcementKind
```

Bases: pybind11_object

```
ALLOW_TYPE_COERCION =
```

```
<TypeConsistencyEnforcementKind.ALLOW_TYPE_COERCION: 1>
```

```
AUTO_TYPE_COERCION =
```

```
<TypeConsistencyEnforcementKind.AUTO_TYPE_COERCION: 2>
```

```
DISALLOW_TYPE_COERCION =
```

```
<TypeConsistencyEnforcementKind.DISALLOW_TYPE_COERCION: 0>
```

```
class TypeConsistencyEnforcementKind
```

Bases: pybind11_object

Members:

DISALLOW_TYPE_COERCION : The DataWriter and the DataReader must support the same data type in order for them to communicate.

ALLOW_TYPE_COERCION : The DataWriter and the DataReader need not support the same data type in order for them to communicate as long as the DataReader's type is assignable from the DataWriter's type.

AUTO_TYPE_COERCION : [default] This AUTO value will be applied as `TypeConsistencyKind.DISALLOW_TYPE_COERCION` when the data type is annotated with `@transfer_mode(SHMEM_REF)` while using C, Traditional C++, or Modern C++ APIs. In all other cases, this AUTO value will be applied as `TypeConsistencyKind.ALLOW_TYPE_COERCION`.

ALLOW_TYPE_COERCION =

`<TypeConsistencyEnforcementKind.ALLOW_TYPE_COERCION: 1>`

AUTO_TYPE_COERCION =

`<TypeConsistencyEnforcementKind.AUTO_TYPE_COERCION: 2>`

DISALLOW_TYPE_COERCION =

`<TypeConsistencyEnforcementKind.DISALLOW_TYPE_COERCION: 0>`

`__eq__` (*self: object, other: object*) → bool

`__getstate__` (*self: object*) → int

`__hash__` (*self: object*) → int

`__index__` (*self: rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind*) →
int

`__init__` (*self: rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind, value: int*) →
None

`__int__` (*self: rti.connextdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind*) →
int

`__members__` = {'ALLOW_TYPE_COERCION':
`<TypeConsistencyEnforcementKind.ALLOW_TYPE_COERCION: 1>`,
'AUTO_TYPE_COERCION':
`<TypeConsistencyEnforcementKind.AUTO_TYPE_COERCION: 2>`,
'DISALLOW_TYPE_COERCION':
`<TypeConsistencyEnforcementKind.DISALLOW_TYPE_COERCION: 0>`}

`__module__` = 'rti.connextdds'

`__ne__` (*self: object, other: object*) → bool

`__repr__` (*self: object*) → str

__setstate__ (*self*: rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind, *state*: int) → None

__str__ ()
name(*self*: handle) -> str

property name

property value

__eq__ (*self*: rti.connexdds.TypeConsistencyEnforcementKind, *arg0*: rti.connexdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.TypeConsistencyEnforcementKind, *arg0*: rti.connexdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.TypeConsistencyEnforcementKind, *arg0*: rti.connexdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.TypeConsistencyEnforcementKind) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.TypeConsistencyEnforcementKind, *arg0*: rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.TypeConsistencyEnforcementKind) → *rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind*

Int conversion.

__le__ (*self*: rti.connexdds.TypeConsistencyEnforcementKind, *arg0*: rti.connexdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.TypeConsistencyEnforcementKind, *arg0*: rti.connexdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.TypeConsistencyEnforcementKind, *arg0*: rti.connexdds.TypeConsistencyEnforcementKind) → bool

Apply operator to underlying enumerated values.

`__str__` (*self*: rti.connexdds.TypeConsistencyEnforcementKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.TypeKind

Bases: pybind11_object

AGGREGATION_TYPE = <TypeKind.AGGREGATION_TYPE: 256>

ALIAS_TYPE = <TypeKind.ALIAS_TYPE: 32771>

ANNOTATION_TYPE = <TypeKind.ANNOTATION_TYPE: 128>

ARRAY_TYPE = <TypeKind.ARRAY_TYPE: 33284>

BITSET_TYPE = <TypeKind.BITSET_TYPE: 32770>

BOOLEAN_TYPE = <TypeKind.BOOLEAN_TYPE: 16385>

CHAR_16_TYPE = <TypeKind.CHAR_16_TYPE: 16397>

CHAR_8_TYPE = <TypeKind.CHAR_8_TYPE: 16396>

COLLECTION_TYPE = <TypeKind.COLLECTION_TYPE: 512>

CONSTRUCTED_TYPE = <TypeKind.CONSTRUCTED_TYPE: 32768>

ENUMERATION_TYPE = <TypeKind.ENUMERATION_TYPE: 32769>

FLOAT_128_TYPE = <TypeKind.FLOAT_128_TYPE: 16395>

FLOAT_32_TYPE = <TypeKind.FLOAT_32_TYPE: 16393>

FLOAT_64_TYPE = <TypeKind.FLOAT_64_TYPE: 16394>

INT_16_TYPE = <TypeKind.INT_16_TYPE: 16387>

INT_32_TYPE = <TypeKind.INT_32_TYPE: 16389>

INT_64_TYPE = <TypeKind.INT_64_TYPE: 16391>

MAP_TYPE = <TypeKind.MAP_TYPE: 33287>

NO_TYPE = <TypeKind.NO_TYPE: 0>

PRIMITIVE_TYPE = <TypeKind.PRIMITIVE_TYPE: 16384>

SEQUENCE_TYPE = <TypeKind.SEQUENCE_TYPE: 33285>

```
STRING_TYPE = <TypeKind.STRING_TYPE: 33286>
```

```
STRUCTURE_FWD_DECL_TYPE = <TypeKind.STRUCTURE_FWD_DECL_TYPE: 33035>
```

```
STRUCTURE_TYPE = <TypeKind.STRUCTURE_TYPE: 33033>
```

```
class TypeKind
```

```
    Bases: pybind11_object
```

```
    Members:
```

```
    NO_TYPE : Sentinel indicating no value.
```

```
    PRIMITIVE_TYPE : Primitive type.
```

```
    CONSTRUCTED_TYPE : Constructed type.
```

```
    COLLECTION_TYPE : Collection type.
```

```
    AGGREGATION_TYPE : Aggregation type.
```

```
    ANNOTATION_TYPE : Annotation type.
```

```
    BOOLEAN_TYPE : Boolean type.
```

```
    UINT_8_TYPE : Unsigned 8-bit int type.
```

```
    INT_16_TYPE : Signed 16-bit int type.
```

```
    UINT_16_TYPE : Unsigned 16-bit int type.
```

```
    INT_32_TYPE : Signed 32-bit int type.
```

```
    UINT_32_TYPE : Unsigned 32-bit int type.
```

```
    INT_64_TYPE : Signed 64-bit int type.
```

```
    UINT_64_TYPE : Unsigned 64-bit int type.
```

```
    FLOAT_32_TYPE : 32-bit floating point type.
```

```
    FLOAT_64_TYPE : 64-bit floating point type.
```

```
    FLOAT_128_TYPE : 128-bit floating point type.
```

```
    CHAR_8_TYPE : 8-bit character type.
```

```
    CHAR_16_TYPE : 16-bit character type.
```

```
    ENUMERATION_TYPE : Enumeration type.
```

```
    BITSET_TYPE : Bitset type.
```

```
    ALIAS_TYPE : Alias type.
```

```
    ARRAY_TYPE : Array type.
```

```
    SEQUENCE_TYPE : Sequence type.
```

```
    STRING_TYPE : String type.
```

```
    WSTRING_TYPE : Wide character string type.
```

MAP_TYPE : Map type

UNION_TYPE : Union type.

STRUCTURE_TYPE : Structure type.

UNION_FWD_DECL_TYPE : Forward declaration of a union type.

STRUCTURE_FWD_DECL_TYPE : Forward declaration of structure type.

AGGREGATION_TYPE = <TypeKind.AGGREGATION_TYPE: 256>

ALIAS_TYPE = <TypeKind.ALIAS_TYPE: 32771>

ANNOTATION_TYPE = <TypeKind.ANNOTATION_TYPE: 128>

ARRAY_TYPE = <TypeKind.ARRAY_TYPE: 33284>

BITSET_TYPE = <TypeKind.BITSET_TYPE: 32770>

BOOLEAN_TYPE = <TypeKind.BOOLEAN_TYPE: 16385>

CHAR_16_TYPE = <TypeKind.CHAR_16_TYPE: 16397>

CHAR_8_TYPE = <TypeKind.CHAR_8_TYPE: 16396>

COLLECTION_TYPE = <TypeKind.COLLECTION_TYPE: 512>

CONSTRUCTED_TYPE = <TypeKind.CONSTRUCTED_TYPE: 32768>

ENUMERATION_TYPE = <TypeKind.ENUMERATION_TYPE: 32769>

FLOAT_128_TYPE = <TypeKind.FLOAT_128_TYPE: 16395>

FLOAT_32_TYPE = <TypeKind.FLOAT_32_TYPE: 16393>

FLOAT_64_TYPE = <TypeKind.FLOAT_64_TYPE: 16394>

INT_16_TYPE = <TypeKind.INT_16_TYPE: 16387>

INT_32_TYPE = <TypeKind.INT_32_TYPE: 16389>

INT_64_TYPE = <TypeKind.INT_64_TYPE: 16391>

MAP_TYPE = <TypeKind.MAP_TYPE: 33287>

NO_TYPE = <TypeKind.NO_TYPE: 0>

PRIMITIVE_TYPE = <TypeKind.PRIMITIVE_TYPE: 16384>

SEQUENCE_TYPE = <TypeKind.SEQUENCE_TYPE: 33285>

STRING_TYPE = <TypeKind.STRING_TYPE: 33286>

STRUCTURE_FWD_DECL_TYPE = <TypeKind.STRUCTURE_FWD_DECL_TYPE: 33035>

```
STRUCTURE_TYPE = <TypeKind.STRUCTURE_TYPE: 33033>
UINT_16_TYPE = <TypeKind.UINT_16_TYPE: 16388>
UINT_32_TYPE = <TypeKind.UINT_32_TYPE: 16390>
UINT_64_TYPE = <TypeKind.UINT_64_TYPE: 16392>
UINT_8_TYPE = <TypeKind.UINT_8_TYPE: 16386>
UNION_FWD_DECL_TYPE = <TypeKind.UNION_FWD_DECL_TYPE: 33034>
UNION_TYPE = <TypeKind.UNION_TYPE: 33032>
WSTRING_TYPE = <TypeKind.WSTRING_TYPE: 33288>

__eq__(self: object, other: object) → bool
__getstate__(self: object) → int
__hash__(self: object) → int
__index__(self: rti.connexdds.TypeKind.TypeKind) → int
__init__(self: rti.connexdds.TypeKind.TypeKind, value: int) → None
__int__(self: rti.connexdds.TypeKind.TypeKind) → int
```

```

__members__ = {'AGGREGATION_TYPE': <TypeKind.AGGREGATION_TYPE:
256>, 'ALIAS_TYPE': <TypeKind.ALIAS_TYPE: 32771>,
'ANNOTATION_TYPE': <TypeKind.ANNOTATION_TYPE: 128>,
'ARRAY_TYPE': <TypeKind.ARRAY_TYPE: 33284>, 'BITSET_TYPE':
<TypeKind.BITSET_TYPE: 32770>, 'BOOLEAN_TYPE':
<TypeKind.BOOLEAN_TYPE: 16385>, 'CHAR_16_TYPE':
<TypeKind.CHAR_16_TYPE: 16397>, 'CHAR_8_TYPE':
<TypeKind.CHAR_8_TYPE: 16396>, 'COLLECTION_TYPE':
<TypeKind.COLLECTION_TYPE: 512>, 'CONSTRUCTED_TYPE':
<TypeKind.CONSTRUCTED_TYPE: 32768>, 'ENUMERATION_TYPE':
<TypeKind.ENUMERATION_TYPE: 32769>, 'FLOAT_128_TYPE':
<TypeKind.FLOAT_128_TYPE: 16395>, 'FLOAT_32_TYPE':
<TypeKind.FLOAT_32_TYPE: 16393>, 'FLOAT_64_TYPE':
<TypeKind.FLOAT_64_TYPE: 16394>, 'INT_16_TYPE':
<TypeKind.INT_16_TYPE: 16387>, 'INT_32_TYPE':
<TypeKind.INT_32_TYPE: 16389>, 'INT_64_TYPE':
<TypeKind.INT_64_TYPE: 16391>, 'MAP_TYPE': <TypeKind.MAP_TYPE:
33287>, 'NO_TYPE': <TypeKind.NO_TYPE: 0>, 'PRIMITIVE_TYPE':
<TypeKind.PRIMITIVE_TYPE: 16384>, 'SEQUENCE_TYPE':
<TypeKind.SEQUENCE_TYPE: 33285>, 'STRING_TYPE':
<TypeKind.STRING_TYPE: 33286>, 'STRUCTURE_FWD_DECL_TYPE':
<TypeKind.STRUCTURE_FWD_DECL_TYPE: 33035>, 'STRUCTURE_TYPE':
<TypeKind.STRUCTURE_TYPE: 33033>, 'UINT_16_TYPE':
<TypeKind.UINT_16_TYPE: 16388>, 'UINT_32_TYPE':
<TypeKind.UINT_32_TYPE: 16390>, 'UINT_64_TYPE':
<TypeKind.UINT_64_TYPE: 16392>, 'UINT_8_TYPE':
<TypeKind.UINT_8_TYPE: 16386>, 'UNION_FWD_DECL_TYPE':
<TypeKind.UNION_FWD_DECL_TYPE: 33034>, 'UNION_TYPE':
<TypeKind.UNION_TYPE: 33032>, 'WSTRING_TYPE':
<TypeKind.WSTRING_TYPE: 33288>}

```

```
__module__ = 'rti.connexdds'
```

```
__ne__(self: object, other: object) → bool
```

```
__repr__(self: object) → str
```

```
__setstate__(self: rti.connexdds.TypeKind.TypeKind, state: int) → None
```

```
__str__()
```

```
name(self: handle) -> str
```

```
property name
```

```
property value
```

```
UINT_16_TYPE = <TypeKind.UINT_16_TYPE: 16388>
```

```
UINT_32_TYPE = <TypeKind.UINT_32_TYPE: 16390>
```

`UINT_64_TYPE = <TypeKind.UINT_64_TYPE: 16392>`

`UINT_8_TYPE = <TypeKind.UINT_8_TYPE: 16386>`

`UNION_FWD_DECL_TYPE = <TypeKind.UNION_FWD_DECL_TYPE: 33034>`

`UNION_TYPE = <TypeKind.UNION_TYPE: 33032>`

`WSTRING_TYPE = <TypeKind.WSTRING_TYPE: 33288>`

`__eq__ (self: rti.connexdds.TypeKind, arg0: rti.connexdds.TypeKind) → bool`

Apply operator to underlying enumerated values.

`__ge__ (self: rti.connexdds.TypeKind, arg0: rti.connexdds.TypeKind) → bool`

Apply operator to underlying enumerated values.

`__gt__ (self: rti.connexdds.TypeKind, arg0: rti.connexdds.TypeKind) → bool`

Apply operator to underlying enumerated values.

`__hash__ = None`

`__init__ (*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connexdds.TypeKind) -> None`

Initializes enum to 0.

2. `__init__(self: rti.connexdds.TypeKind, arg0: rti.connexdds.TypeKind.TypeKind) -> None`

Copy constructor.

`__int__ (self: rti.connexdds.TypeKind) → rti.connexdds.TypeKind.TypeKind`

Int conversion.

`__le__ (self: rti.connexdds.TypeKind, arg0: rti.connexdds.TypeKind) → bool`

Apply operator to underlying enumerated values.

`__lt__ (self: rti.connexdds.TypeKind, arg0: rti.connexdds.TypeKind) → bool`

Apply operator to underlying enumerated values.

`__module__ = 'rti.connexdds'`

`__ne__ (self: rti.connexdds.TypeKind, arg0: rti.connexdds.TypeKind) → bool`

Apply operator to underlying enumerated values.

`__str__ (self: rti.connexdds.TypeKind) → str`

String conversion.

property underlying

Retrieves the actual enumerated value.

class `rti.connexdds.TypeSupport`

Bases: `pybind11_object`

__eq__ (*self*: rti.connexdds.TypeSupport, *arg0*: rti.connexdds.TypeSupport) → bool
 Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.TypeSupport) → None
 Creates the default policy.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.TypeSupport, *arg0*: rti.connexdds.TypeSupport) → bool
 Test for inequality.

property cdr_padding_kind

Determines whether or not the padding bytes will be set to zero during CDR serialization.

rti.connexdds.**ULongLongSeq**
 alias of *Uint64Seq*

rti.connexdds.**ULongLongType**
 alias of *Uint64Type*

rti.connexdds.**ULongSeq**
 alias of *Uint32Seq*

rti.connexdds.**ULongType**
 alias of *Uint32Type*

rti.connexdds.**UShortType**
 alias of *Uint16Type*

class rti.connexdds.**Uint16Seq**

Bases: pybind11_object

__add__ (*self*: rti.connexdds.Uint16Seq, *arg0*: rti.connexdds.Uint16Seq) →
rti.connexdds.Uint16Seq

__bool__ (*self*: rti.connexdds.Uint16Seq) → bool
 Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.Uint16Seq, *x*: int) → bool
 Return true the container contains x

__delitem__ (**args*, ***kwargs*)
 Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.Uint16Seq, *arg0*: int) -> None

Delete the list elements at index *i*

2. **__delitem__**(*self*: rti.connexdds.Uint16Seq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.Uint16Seq, *arg0*: rti.connexdds.Uint16Seq) → bool

__getitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.Uint16Seq, *s*: slice) -> rti.connexdds.Uint16Seq
Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.Uint16Seq, *arg0*: int) -> int

__getstate__ (*self*: rti.connexdds.Uint16Seq) → bytes

__hash__ = None

__iadd__ (*self*: rti.connexdds.Uint16Seq, *arg0*: rti.connexdds.Uint16Seq) → *rti.connexdds.Uint16Seq*

__imul__ (*self*: rti.connexdds.Uint16Seq, *arg0*: int) → *rti.connexdds.Uint16Seq*

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: rti.connexdds.Uint16Seq, *arg0*: buffer) -> None
2. **__init__**(*self*: rti.connexdds.Uint16Seq) -> None
3. **__init__**(*self*: rti.connexdds.Uint16Seq, *arg0*: rti.connexdds.Uint16Seq) -> None
Copy constructor
4. **__init__**(*self*: rti.connexdds.Uint16Seq, *arg0*: Iterable) -> None
5. **__init__**(*self*: rti.connexdds.Uint16Seq, *arg0*: int) -> None

__iter__ (*self*: rti.connexdds.Uint16Seq) → Iterator

__len__ (*self*: rti.connexdds.Uint16Seq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.Uint16Seq, *arg0*: int) → *rti.connexdds.Uint16Seq*

__ne__ (*self*: rti.connexdds.Uint16Seq, *arg0*: rti.connexdds.Uint16Seq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (*self*: rti.connexdds.Uint16Seq) → str
Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.Uint16Seq, *arg0*: int) → *rti.connexdds.Uint16Seq*

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.Uint16Seq, arg0: int, arg1: int) -> None`
2. `__setitem__(self: rti.connextdds.Uint16Seq, arg0: slice, arg1: rti.connextdds.Uint16Seq) -> None`

Assign list elements using a slice object

__setstate__ (self: rti.connextdds.Uint16Seq, arg0: bytes) → None

append (self: rti.connextdds.Uint16Seq, x: int) → None

Add an item to the end of the list

clear (self: rti.connextdds.Uint16Seq) → None

Clear the contents

count (self: rti.connextdds.Uint16Seq, x: int) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connextdds.Uint16Seq, L: rti.connextdds.Uint16Seq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.Uint16Seq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.Uint16Seq, i: int, x: int) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.Uint16Seq) -> int`

Remove and return the last item

2. `pop(self: rti.connextdds.Uint16Seq, i: int) -> int`

Remove and return the item at index i

remove (self: rti.connextdds.Uint16Seq, x: int) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

resize (self: rti.connextdds.Uint16Seq, arg0: int) → None

resizes the vector to the given size

class `rti.connextdds.Uint16Type`

Bases: *DynamicType*

`__eq__` (*self*: rti.connexdds.Uint16Type, *arg0*: rti.connexdds.Uint16Type) → bool

Test for equality.

`__hash__` = None

`__init__` (*self*: rti.connexdds.Uint16Type) → None

Get the singleton for Uint16Type

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.Uint16Type, *arg0*: rti.connexdds.Uint16Type) → bool

Test for inequality.

class rti.connexdds.Uint32Seq

Bases: pybind11_object

`__add__` (*self*: rti.connexdds.Uint32Seq, *arg0*: rti.connexdds.Uint32Seq) →
rti.connexdds.Uint32Seq

`__bool__` (*self*: rti.connexdds.Uint32Seq) → bool

Check whether the list is nonempty

`__contains__` (*self*: rti.connexdds.Uint32Seq, *x*: int) → bool

Return true the container contains *x*

`__delitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__delitem__`(*self*: rti.connexdds.Uint32Seq, *arg0*: int) -> None

Delete the list elements at index *i*

2. `__delitem__`(*self*: rti.connexdds.Uint32Seq, *arg0*: slice) -> None

Delete list elements using a slice object

`__eq__` (*self*: rti.connexdds.Uint32Seq, *arg0*: rti.connexdds.Uint32Seq) → bool

`__getitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__getitem__`(*self*: rti.connexdds.Uint32Seq, *s*: slice) -> rti.connexdds.Uint32Seq

Retrieve list elements using a slice object

2. `__getitem__`(*self*: rti.connexdds.Uint32Seq, *arg0*: int) -> int

`__getstate__` (*self*: rti.connexdds.Uint32Seq) → bytes

`__hash__` = None

`__iadd__` (*self*: rti.connexdds.Uint32Seq, *arg0*: rti.connexdds.Uint32Seq) →
rti.connexdds.Uint32Seq

`__imul__` (*self*: rti.connexdds.Uint32Seq, *arg0*: int) → *rti.connexdds.Uint32Seq*

`__init__` (**args*, ***kwargs*)
Overloaded function.

1. `__init__(self: rti.connexdds.Uint32Seq, arg0: buffer) -> None`
2. `__init__(self: rti.connexdds.Uint32Seq) -> None`
3. `__init__(self: rti.connexdds.Uint32Seq, arg0: rti.connexdds.Uint32Seq) -> None`

Copy constructor

4. `__init__(self: rti.connexdds.Uint32Seq, arg0: Iterable) -> None`
5. `__init__(self: rti.connexdds.Uint32Seq, arg0: int) -> None`

`__iter__` (*self*: rti.connexdds.Uint32Seq) → Iterator

`__len__` (*self*: rti.connexdds.Uint32Seq) → int

`__module__` = `'rti.connexdds'`

`__mul__` (*self*: rti.connexdds.Uint32Seq, *arg0*: int) → *rti.connexdds.Uint32Seq*

`__ne__` (*self*: rti.connexdds.Uint32Seq, *arg0*: rti.connexdds.Uint32Seq) → bool

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` = `<capsule object NULL>`

`__repr__` (*self*: rti.connexdds.Uint32Seq) → str
Return the canonical string representation of this list.

`__rmul__` (*self*: rti.connexdds.Uint32Seq, *arg0*: int) → *rti.connexdds.Uint32Seq*

`__setitem__` (**args*, ***kwargs*)
Overloaded function.

1. `__setitem__(self: rti.connexdds.Uint32Seq, arg0: int, arg1: int) -> None`
2. `__setitem__(self: rti.connexdds.Uint32Seq, arg0: slice, arg1: rti.connexdds.Uint32Seq) -> None`

Assign list elements using a slice object

`__setstate__` (*self*: rti.connexdds.Uint32Seq, *arg0*: bytes) → None

`append` (*self*: rti.connexdds.Uint32Seq, *x*: int) → None
Add an item to the end of the list

`clear` (*self*: rti.connexdds.Uint32Seq) → None
Clear the contents

`count` (*self*: rti.connexdds.Uint32Seq, *x*: int) → int
Return the number of times *x* appears in the list

extend (*args, **kwargs)

Overloaded function.

1. extend(self: rti.connextdds.Uint32Seq, L: rti.connextdds.Uint32Seq) -> None

Extend the list by appending all the items in the given list

2. extend(self: rti.connextdds.Uint32Seq, L: Iterable) -> None

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.Uint32Seq, i: int, x: int) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. pop(self: rti.connextdds.Uint32Seq) -> int

Remove and return the last item

2. pop(self: rti.connextdds.Uint32Seq, i: int) -> int

Remove and return the item at index i

remove (self: rti.connextdds.Uint32Seq, x: int) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

resize (self: rti.connextdds.Uint32Seq, arg0: int) → None

resizes the vector to the given size

class rti.connextdds.Uint32Type

Bases: *DynamicType*

__eq__ (self: rti.connextdds.Uint32Type, arg0: rti.connextdds.Uint32Type) → bool

Test for equality.

__hash__ = None

__init__ (self: rti.connextdds.Uint32Type) → None

Get the singleton for Uint32Type

__module__ = 'rti.connextdds'

__ne__ (self: rti.connextdds.Uint32Type, arg0: rti.connextdds.Uint32Type) → bool

Test for inequality.

class rti.connextdds.Uint64Seq

Bases: *pybind11_object*

__add__ (self: rti.connextdds.Uint64Seq, arg0: rti.connextdds.Uint64Seq) →
rti.connextdds.Uint64Seq

__bool__ (self: rti.connextdds.Uint64Seq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.Uint64Seq, *x*: int) → bool
Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.Uint64Seq, *arg0*: int) -> None
Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connexdds.Uint64Seq, *arg0*: slice) -> None
Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.Uint64Seq, *arg0*: rti.connexdds.Uint64Seq) → bool

__getitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.Uint64Seq, *s*: slice) -> rti.connexdds.Uint64Seq
Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.Uint64Seq, *arg0*: int) -> int

__getstate__ (*self*: rti.connexdds.Uint64Seq) → bytes

__hash__ = None

__iadd__ (*self*: rti.connexdds.Uint64Seq, *arg0*: rti.connexdds.Uint64Seq) → *rti.connexdds.Uint64Seq*

__imul__ (*self*: rti.connexdds.Uint64Seq, *arg0*: int) → *rti.connexdds.Uint64Seq*

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: rti.connexdds.Uint64Seq, *arg0*: buffer) -> None
2. **__init__**(*self*: rti.connexdds.Uint64Seq) -> None
3. **__init__**(*self*: rti.connexdds.Uint64Seq, *arg0*: rti.connexdds.Uint64Seq) -> None
Copy constructor
4. **__init__**(*self*: rti.connexdds.Uint64Seq, *arg0*: Iterable) -> None
5. **__init__**(*self*: rti.connexdds.Uint64Seq, *arg0*: int) -> None

__iter__ (*self*: rti.connexdds.Uint64Seq) → Iterator

__len__ (*self*: rti.connexdds.Uint64Seq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.Uint64Seq, *arg0*: int) → *rti.connexdds.Uint64Seq*

__ne__ (*self*: rti.connexdds.Uint64Seq, *arg0*: rti.connexdds.Uint64Seq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (*self*: rti.connexdds.Uint64Seq) → str

Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.Uint64Seq, *arg0*: int) → rti.connexdds.Uint64Seq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.Uint64Seq, *arg0*: int, *arg1*: int) -> None
2. **__setitem__**(*self*: rti.connexdds.Uint64Seq, *arg0*: slice, *arg1*: rti.connexdds.Uint64Seq) -> None

Assign list elements using a slice object

__setstate__ (*self*: rti.connexdds.Uint64Seq, *arg0*: bytes) → None

append (*self*: rti.connexdds.Uint64Seq, *x*: int) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.Uint64Seq) → None

Clear the contents

count (*self*: rti.connexdds.Uint64Seq, *x*: int) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.Uint64Seq, *L*: rti.connexdds.Uint64Seq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.Uint64Seq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.Uint64Seq, *i*: int, *x*: int) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.Uint64Seq) -> int

Remove and return the last item

2. **pop**(*self*: rti.connexdds.Uint64Seq, *i*: int) -> int

Remove and return the item at index *i*

remove (*self*: rti.connexdds.Uint64Seq, *x*: int) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

resize (*self*: rti.connexdds.Uint64Seq, *arg0*: int) → None

resizes the vector to the given size

class rti.connexdds.**Uint64Type**

Bases: *DynamicType*

__eq__ (*self*: rti.connexdds.Uint64Type, *arg0*: rti.connexdds.Uint64Type) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.Uint64Type) → None

Get the singleton for Uint64Type

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.Uint64Type, *arg0*: rti.connexdds.Uint64Type) → bool

Test for inequality.

class rti.connexdds.**Uint8Seq**

Bases: *pybind11_object*

__add__ (*self*: rti.connexdds.Uint8Seq, *arg0*: rti.connexdds.Uint8Seq) → *rti.connexdds.Uint8Seq*

__bool__ (*self*: rti.connexdds.Uint8Seq) → bool

Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.Uint8Seq, *x*: int) → bool

Return true the container contains x

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.Uint8Seq, *arg0*: int) -> None

Delete the list elements at index i

2. **__delitem__**(*self*: rti.connexdds.Uint8Seq, *arg0*: slice) -> None

Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.Uint8Seq, *arg0*: rti.connexdds.Uint8Seq) → bool

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.Uint8Seq, *s*: slice) -> rti.connexdds.Uint8Seq

Retrieve list elements using a slice object

2. **__getitem__**(*self*: rti.connexdds.Uint8Seq, *arg0*: int) -> int

__getstate__ (*self*: rti.connexdds.Uint8Seq) → bytes

__hash__ = None

__iadd__ (*self*: rti.connexdds.Uint8Seq, *arg0*: rti.connexdds.Uint8Seq) → *rti.connexdds.Uint8Seq*

__imul__ (*self*: rti.connexdds.Uint8Seq, *arg0*: int) → *rti.connexdds.Uint8Seq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.Uint8Seq, *arg0*: buffer) -> None
2. **__init__**(*self*: rti.connexdds.Uint8Seq) -> None
3. **__init__**(*self*: rti.connexdds.Uint8Seq, *arg0*: rti.connexdds.Uint8Seq) -> None

Copy constructor

4. **__init__**(*self*: rti.connexdds.Uint8Seq, *arg0*: Iterable) -> None
5. **__init__**(*self*: rti.connexdds.Uint8Seq, *arg0*: int) -> None

__iter__ (*self*: rti.connexdds.Uint8Seq) → Iterator

__len__ (*self*: rti.connexdds.Uint8Seq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.Uint8Seq, *arg0*: int) → *rti.connexdds.Uint8Seq*

__ne__ (*self*: rti.connexdds.Uint8Seq, *arg0*: rti.connexdds.Uint8Seq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__repr__ (*self*: rti.connexdds.Uint8Seq) → str

Return the canonical string representation of this list.

__rmul__ (*self*: rti.connexdds.Uint8Seq, *arg0*: int) → *rti.connexdds.Uint8Seq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.Uint8Seq, *arg0*: int, *arg1*: int) -> None
2. **__setitem__**(*self*: rti.connexdds.Uint8Seq, *arg0*: slice, *arg1*: rti.connexdds.Uint8Seq) -> None

Assign list elements using a slice object

__setstate__ (*self*: rti.connexdds.Uint8Seq, *arg0*: bytes) → None

append (*self*: rti.connexdds.Uint8Seq, *x*: int) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.Uint8Seq) → None

Clear the contents

count (*self*: rti.connexdds.Uint8Seq, *x*: int) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. extend(*self*: rti.connexdds.Uint8Seq, *L*: rti.connexdds.Uint8Seq) → None

Extend the list by appending all the items in the given list

2. extend(*self*: rti.connexdds.Uint8Seq, *L*: Iterable) → None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.Uint8Seq, *i*: int, *x*: int) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. pop(*self*: rti.connexdds.Uint8Seq) → int

Remove and return the last item

2. pop(*self*: rti.connexdds.Uint8Seq, *i*: int) → int

Remove and return the item at index *i*

remove (*self*: rti.connexdds.Uint8Seq, *x*: int) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

resize (*self*: rti.connexdds.Uint8Seq, *arg0*: int) → None

resizes the vector to the given size

class rti.connexdds.Uint8Type

Bases: *DynamicType*

__eq__ (*self*: rti.connexdds.Uint8Type, *arg0*: rti.connexdds.Uint8Type) → bool

Test for equality.

__hash__ = None

__init__ (*self*: rti.connexdds.Uint8Type) → None

Get the singleton for Uint8Type

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.Uint8Type, *arg0*: rti.connexdds.Uint8Type) → bool

Test for inequality.

class rti.connexdds.UnidimensionalCollectionType

Bases: *CollectionType*

UNBOUNDED = 2147483647

__eq__ (*self*: rti.connexdds.UnidimensionalCollectionType, *arg0*: rti.connexdds.UnidimensionalCollectionType) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.UnidimensionalCollectionType, *arg0*: rti.connexdds.UnidimensionalCollectionType) → bool

Test for inequality.

property bounds

Gets the maximum length of this collection.

class rti.connexdds.**UnionMember**

Bases: pybind11_object

DEFAULT_LABEL = 1073741825

INVALID_ID = 2147483647

__eq__ (*self*: rti.connexdds.UnionMember, *arg0*: rti.connexdds.UnionMember) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.UnionMember, *name*: str, *dynamic_type*: rti.connexdds.DynamicType, *label*: int) -> None

Creates a union member with a name, type, and selected by a single label.

2. **__init__**(*self*: rti.connexdds.UnionMember, *name*: str, *dynamic_type*: rti.connexdds.DynamicType, *labels*: rti.connexdds.Int32Seq) -> None

Create a union member with a name, type, and selected by one or more labels.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.UnionMember, *arg0*: rti.connexdds.UnionMember) → bool

Test for inequality.

property has_id

Indicates if union member has an ID annotation.

property id

Returns the ID annotation of this member, or INVALID_ID if this member has no ID.

property label_count

Gets the number of labels that select this member.

property labels

The labels that select this member.

property name

The union member's name.

property pointer

Boolean flag for pointer status of union member.

property type

Gets the union member's type.

class `rti.connexdds.UnionMemberSeq`

Bases: `pybind11_object`

__add__ (*self*: `rti.connexdds.UnionMemberSeq`, *arg0*: `rti.connexdds.UnionMemberSeq`) → `rti.connexdds.UnionMemberSeq`

__bool__ (*self*: `rti.connexdds.UnionMemberSeq`) → `bool`

Check whether the list is nonempty

__contains__ (*self*: `rti.connexdds.UnionMemberSeq`, *x*: `rti.connexdds.UnionMember`) → `bool`

Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__delitem__**(*self*: `rti.connexdds.UnionMemberSeq`, *arg0*: `int`) -> `None`

Delete the list elements at index *i*

2. **__delitem__**(*self*: `rti.connexdds.UnionMemberSeq`, *arg0*: `slice`) -> `None`

Delete list elements using a slice object

__eq__ (*self*: `rti.connexdds.UnionMemberSeq`, *arg0*: `rti.connexdds.UnionMemberSeq`) → `bool`

__getitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__getitem__**(*self*: `rti.connexdds.UnionMemberSeq`, *s*: `slice`) -> `rti.connexdds.UnionMemberSeq`

Retrieve list elements using a slice object

2. **__getitem__**(*self*: `rti.connexdds.UnionMemberSeq`, *arg0*: `int`) -> `rti.connexdds.UnionMember`

__hash__ = `None`

__iadd__ (*self*: rti.connexdds.UnionMemberSeq, *arg0*: rti.connexdds.UnionMemberSeq) → *rti.connexdds.UnionMemberSeq*

__imul__ (*self*: rti.connexdds.UnionMemberSeq, *arg0*: int) → *rti.connexdds.UnionMemberSeq*

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.UnionMemberSeq) -> None
2. **__init__**(*self*: rti.connexdds.UnionMemberSeq, *arg0*: rti.connexdds.UnionMemberSeq) -> None

Copy constructor

3. **__init__**(*self*: rti.connexdds.UnionMemberSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.UnionMemberSeq) → Iterator

__len__ (*self*: rti.connexdds.UnionMemberSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.UnionMemberSeq, *arg0*: int) → *rti.connexdds.UnionMemberSeq*

__ne__ (*self*: rti.connexdds.UnionMemberSeq, *arg0*: rti.connexdds.UnionMemberSeq) → bool

__rmul__ (*self*: rti.connexdds.UnionMemberSeq, *arg0*: int) → *rti.connexdds.UnionMemberSeq*

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.UnionMemberSeq, *arg0*: int, *arg1*: rti.connexdds.UnionMember) -> None
2. **__setitem__**(*self*: rti.connexdds.UnionMemberSeq, *arg0*: slice, *arg1*: rti.connexdds.UnionMemberSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.UnionMemberSeq, *x*: rti.connexdds.UnionMember) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.UnionMemberSeq) → None

Clear the contents

count (*self*: rti.connexdds.UnionMemberSeq, *x*: rti.connexdds.UnionMember) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.UnionMemberSeq, *L*: rti.connexdds.UnionMemberSeq) -> None

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.UnionMemberSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (*self*: rti.connextdds.UnionMemberSeq, *i*: int, *x*: rti.connextdds.UnionMember) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.UnionMemberSeq) -> rti.connextdds.UnionMember`

Remove and return the last item

2. `pop(self: rti.connextdds.UnionMemberSeq, i: int) -> rti.connextdds.UnionMember`

Remove and return the item at index *i*

remove (*self*: rti.connextdds.UnionMemberSeq, *x*: rti.connextdds.UnionMember) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

class rti.connextdds.UnionType

Bases: *ACTUnionMember*

__eq__ (*self*: rti.connextdds.UnionType, *arg0*: rti.connextdds.UnionType) → bool

Test for equality.

__hash__ = None

__init__ (*args, **kwargs)

Overloaded function.

1. `__init__(self: rti.connextdds.UnionType, name: str, discriminator_type: rti.connextdds.DynamicType) -> None`

Creates an empty Union.

2. `__init__(self: rti.connextdds.UnionType, name: str, discriminator_type: rti.connextdds.DynamicType, members: rti.connextdds.UnionMemberSeq) -> None`

Creates a Union with the specified UnionMembers.

3. `__init__(self: rti.connextdds.UnionType, type: rti.connextdds.DynamicType) -> None`

Cast a DynamicType to a UnionType.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.UnionType, *arg0*: rti.connextdds.UnionType) → bool

Test for inequality.

add_members (*self*: rti.connextdds.UnionType, *members*: rti.connextdds.UnionMemberSeq) → *rti.connextdds.UnionType*

Adds all the members of a container at the end.

property discriminator

The union discriminator type.

property extensibility_kind

Union's extensibility kind.

find_member_by_id (*self*: rti.connexdds.UnionType, *id*: int) → int

Gets the index of the member selected by an ID.

find_member_by_label (*self*: rti.connexdds.UnionType, *label*: int) → int

Gets the index of the member selected by a label.

exception rti.connexdds.UnsupportedError

Bases: *Exception*

__module__ = 'rti.connexdds'

class rti.connexdds.UserData

Bases: *pybind11_object*

__eq__ (*self*: rti.connexdds.UserData, *arg0*: rti.connexdds.UserData) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.UserData) -> None

Creates an instance with an empty sequence of bytes.

2. **__init__**(*self*: rti.connexdds.UserData, *data*: rti.connexdds.Uint8Seq) -> None

Creates an instance with a sequence of bytes.

__iter__ (*self*: rti.connexdds.UserData) → Iterator

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.UserData, *arg0*: rti.connexdds.UserData) → bool

Test for inequality.

property value

The user data.

class rti.connexdds.UserDataSample

Bases: *pybind11_object*

An alias for a Python object representing a data sample of an IDL type

__init__ (**args*, ***kwargs*)

__module__ = 'rti.connexdds'

`rti.connexdds.UshortSeq`

alias of `Uint16Seq`

class `rti.connexdds.VendorId`

Bases: `pybind11_object`

`__eq__` (*self*: `rti.connexdds.VendorId`, *arg0*: `rti.connexdds.VendorId`) → bool

Test for equality.

`__hash__` = None

`__init__` (*self*: `rti.connexdds.VendorId`) → None

Creates unknown vendor ID.

`__module__` = `'rti.connexdds'`

`__ne__` (*self*: `rti.connexdds.VendorId`, *arg0*: `rti.connexdds.VendorId`) → bool

Test for inequality.

`unknown` = `<rti.connexdds.VendorId object>`

property value

Provides access to the bytes that represent the vendor id.

class `rti.connexdds.Verbosity`

Bases: `pybind11_object`

`EXCEPTION` = `<Verbosity.EXCEPTION: 3>`

`SILENT` = `<Verbosity.SILENT: 0>`

`STATUS_ALL` = `<Verbosity.STATUS_ALL: 63>`

`STATUS_LOCAL` = `<Verbosity.STATUS_LOCAL: 15>`

`STATUS_REMOTE` = `<Verbosity.STATUS_REMOTE: 31>`

class `Verbosity`

Bases: `pybind11_object`

Members:

`SILENT` : No further output will be logged.

`EXCEPTION` : Only error messages will be logged.

An error indicates something wrong in the functioning of RTI Connex. The most common cause of errors is incorrect configuration.

`WARNING` : Both error and warning messages will be logged.

A warning indicates that RTI Connex is taking an action that may or may not be what you intended. Some configuration information is also logged at this verbosity to aid in debugging.

STATUS_LOCAL : Errors, warnings, and verbose information about the lifecycles of local RTI Connex objects will be logged.

STATUS_REMOTE : Errors, warnings, and verbose information about the lifecycles of remote RTI Connex objects will be logged.

STATUS_ALL : Errors, warnings, verbose information about the lifecycles of local and remote RTI Connex objects, and periodic information about RTI Connex threads will be logged.

```

EXCEPTION = <Verbosity.EXCEPTION: 3>

SILENT = <Verbosity.SILENT: 0>

STATUS_ALL = <Verbosity.STATUS_ALL: 63>

STATUS_LOCAL = <Verbosity.STATUS_LOCAL: 15>

STATUS_REMOTE = <Verbosity.STATUS_REMOTE: 31>

WARNING = <Verbosity.WARNING: 7>

__eq__ (self: object, other: object) → bool

__getstate__ (self: object) → int

__hash__ (self: object) → int

__index__ (self: rti.connexdds.Verbosity.Verbosity) → int

__init__ (self: rti.connexdds.Verbosity.Verbosity, value: int) → None

__int__ (self: rti.connexdds.Verbosity.Verbosity) → int

__members__ = {'EXCEPTION': <Verbosity.EXCEPTION: 3>, 'SILENT':
<Verbosity.SILENT: 0>, 'STATUS_ALL': <Verbosity.STATUS_ALL:
63>, 'STATUS_LOCAL': <Verbosity.STATUS_LOCAL: 15>,
'STATUS_REMOTE': <Verbosity.STATUS_REMOTE: 31>, 'WARNING':
<Verbosity.WARNING: 7>}

__module__ = 'rti.connexdds'

__ne__ (self: object, other: object) → bool

__repr__ (self: object) → str

__setstate__ (self: rti.connexdds.Verbosity.Verbosity, state: int) → None

__str__ ()
    name(self: handle) -> str

property name

property value

```

WARNING = <Verbosity.WARNING: 7>

__eq__ (*self*: rti.connexdds.Verboseity, *arg0*: rti.connexdds.Verboseity) → bool
Apply operator to underlying enumerated values.

__ge__ (*self*: rti.connexdds.Verboseity, *arg0*: rti.connexdds.Verboseity) → bool
Apply operator to underlying enumerated values.

__gt__ (*self*: rti.connexdds.Verboseity, *arg0*: rti.connexdds.Verboseity) → bool
Apply operator to underlying enumerated values.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.Verboseity) -> None

Initializes enum to 0.

2. **__init__**(*self*: rti.connexdds.Verboseity, *arg0*: rti.connexdds.Verboseity.Verboseity) -> None

Copy constructor.

__int__ (*self*: rti.connexdds.Verboseity) → *rti.connexdds.Verboseity.Verboseity*
Int conversion.

__le__ (*self*: rti.connexdds.Verboseity, *arg0*: rti.connexdds.Verboseity) → bool
Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.Verboseity, *arg0*: rti.connexdds.Verboseity) → bool
Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.Verboseity, *arg0*: rti.connexdds.Verboseity) → bool
Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.Verboseity) → str
String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.ViewState

Bases: pybind11_object

ANY = <rti.connexdds.ViewState object>

NEW_VIEW = <rti.connexdds.ViewState object>

NOT_NEW_VIEW = <rti.connexdds.ViewState object>

__and__ (*self*: rti.connexdds.ViewState, *arg0*: rti.connexdds.ViewState) → *rti.connexdds.ViewState*
Bitwise logical AND of masks.

__bool__ (*self*: rti.connexdds.ViewState) → *rti.connexdds.ViewState*
Test if any bits are set.

__contains__ (*self*: rti.connexdds.ViewState, *arg0*: rti.connexdds.ViewState) → bool

__eq__ (*self*: rti.connexdds.ViewState, *arg0*: rti.connexdds.ViewState) → bool
Compare masks for equality.

__getitem__ (*self*: rti.connexdds.ViewState, *arg0*: int) → bool
Get individual mask bit.

__hash__ = None

__iand__ (*self*: rti.connexdds.ViewState, *arg0*: rti.connexdds.ViewState) → *rti.connexdds.ViewState*
Set mask to logical AND with another mask.

__ilshift__ (*self*: rti.connexdds.ViewState, *arg0*: int) → *rti.connexdds.ViewState*
Left shift bits in mask.

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: rti.connexdds.ViewState) -> None
Create a ViewState with no bits set.
2. **__init__**(*self*: rti.connexdds.ViewState, *value*: int) -> None
Creates a mask from the bits in an integer.

__int__ (*self*: rti.connexdds.ViewState) → int
Convert mask to int.

__ior__ (*self*: rti.connexdds.ViewState, *arg0*: rti.connexdds.ViewState) → *rti.connexdds.ViewState*
Set mask to logical OR with another mask.

__irshift__ (*self*: rti.connexdds.ViewState, *arg0*: int) → *rti.connexdds.ViewState*
Right shift bits in mask.

__ixor__ (*self*: rti.connexdds.ViewState, *arg0*: rti.connexdds.ViewState) → *rti.connexdds.ViewState*
Set mask to logical XOR with another mask.

__lshift__ (*self*: rti.connexdds.ViewState, *arg0*: int) → *rti.connexdds.ViewState*
Left shift bits in mask.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.ViewState, *arg0*: rti.connexdds.ViewState) → bool
Compare masks for inequality.

__or__ (*self*: rti.connexdds.ViewState, *arg0*: rti.connexdds.ViewState) → *rti.connexdds.ViewState*
Bitwise logical OR of masks.

__rshift__ (*self*: rti.connexdds.ViewState, *arg0*: int) → *rti.connexdds.ViewState*
Right shift bits in mask.

__setitem__ (*self*: rti.connexdds.ViewState, *arg0*: int, *arg1*: bool) → None
Set individual mask bit

__str__ (*self*: rti.connexdds.ViewState) → str

__xor__ (*self*: rti.connexdds.ViewState, *arg0*: rti.connexdds.ViewState) →
rti.connexdds.ViewState
Bitwise logical XOR of masks.

property count

Returns the number of bits set in the mask.

flip (**args*, ***kwargs*)

Overloaded function.

1. flip(*self*: rti.connexdds.ViewState) -> rti.connexdds.ViewState

Flip all bits in the mask.

2. flip(*self*: rti.connexdds.ViewState, *pos*: int) -> rti.connexdds.ViewState

Flip the mask bit at the specified position.

reset (**args*, ***kwargs*)

Overloaded function.

1. reset(*self*: rti.connexdds.ViewState) -> rti.connexdds.ViewState

Clear all bits in the mask.

2. reset(*self*: rti.connexdds.ViewState, *pos*: int) -> rti.connexdds.ViewState

Clear the mask bit at the specified position.

set (**args*, ***kwargs*)

Overloaded function.

1. set(*self*: rti.connexdds.ViewState) -> rti.connexdds.ViewState

Set all bits in the mask.

2. set(*self*: rti.connexdds.ViewState, *pos*: int, *value*: bool = True) -> rti.connexdds.ViewState

Set the mask bit at the specified position to the provided value (default: true).

property size

Returns the number of bits in the mask type.

test (*self*: rti.connexdds.ViewState, *pos*: int) → bool

Test whether the mask bit at position “pos” is set.

test_all (*self*: rti.connexdds.ViewState) → bool

Test if all bits are set.

test_any (*self*: rti.connexdds.ViewState) → bool

Test if any bits are set.

test_none (*self*: rti.connexdds.ViewState) → bool

Test if none of the bits are set.

class rti.connexdds.WStringType

Bases: *UnidimensionalCollectionType*

__eq__ (*self*: rti.connexdds.WStringType, *arg0*: rti.connexdds.WStringType) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.WStringType, *bounds*: int) -> None

Creates a bounded wide string.

2. **__init__**(*self*: rti.connexdds.WStringType) -> None

Creates an unbounded wide string.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.WStringType, *arg0*: rti.connexdds.WStringType) → bool

Test for inequality.

class rti.connexdds.WaitSet

Bases: *pybind11_object*

__iadd__ (*self*: rti.connexdds.WaitSet, *arg0*: rti.connexdds.ICondition) → *rti.connexdds.WaitSet*

Attach a condition to a WaitSet.

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connexdds.WaitSet) -> None

Create a WaitSet with no conditions attached.

2. **__init__**(*self*: rti.connexdds.WaitSet, *property*: rti.connexdds.WaitSetProperty) -> None

Create a WaitSet with no conditions attached with the specified WaitSetProperty settings.

__isub__ (*self*: rti.connexdds.WaitSet, *arg0*: rti.connexdds.ICondition) → *rti.connexdds.WaitSet*

Detach a condition from a WaitSet.

__module__ = 'rti.connexdds'

attach_condition (*self*: rti.connexdds.WaitSet, *condition*: rti.connexdds.ICondition) → None
Attach a condition to the WaitSet.

property conditions

Get/set the attached conditions for the WaitSet.

detach_all (*self*: rti.connexdds.WaitSet) → None
Detach all conditions from the WaitSet.

detach_condition (*self*: rti.connexdds.WaitSet, *condition*: rti.connexdds.ICondition) → bool
Detach a condition from the WaitSet.

dispatch (**args*, ***kwargs*)

Overloaded function.

1. `dispatch(self: rti.connexdds.WaitSet, timeout: rti.connexdds.Duration) -> None`

Dispatch handlers for triggered conditions attached to this WaitSet with a timeout.

2. `dispatch(self: rti.connexdds.WaitSet) -> None`

Dispatch handlers for triggered conditions attached to this WaitSet with no timeout.

dispatch_async (**args*, ***kwargs*)

Overloaded function.

1. `dispatch_async(self: rti.connexdds.WaitSet, timeout: rti.connexdds.Duration) -> object`

Dispatch handlers for triggered conditions attached to this WaitSet with a timeout. This call is awaitable and only for use with asyncio.

2. `dispatch_async(self: rti.connexdds.WaitSet) -> object`

Dispatch handlers for triggered conditions attached to this WaitSet with no timeout. This call is awaitable and only for use with asyncio.

property property

Get/set the WaitSetProperty settings for the WaitSet.

wait (**args*, ***kwargs*)

Overloaded function.

1. `wait(self: rti.connexdds.WaitSet, timeout: rti.connexdds.Duration) -> rti.connexdds.TriggeredConditions`

Wait for conditions attached to this WaitSet to trigger with a timeout.

2. `wait(self: rti.connexdds.WaitSet) -> rti.connexdds.TriggeredConditions`

Wait indefinitely for conditions attached to this WaitSet to trigger.

wait_async (**args*, ***kwargs*)

Overloaded function.

1. `wait_async(self: rti.connexdds.WaitSet, timeout: rti.connexdds.Duration) -> object`

Wait for conditions attached to this WaitSet to trigger with a timeout. This call is awaitable and only for use with asyncio.

2. `wait_async(self: rti.connextdds.WaitSet) -> object`

Wait indefinitely for conditions attached to this WaitSet to trigger. This call is awaitable and only for use with asyncio.

class `rti.connextdds.WaitSetProperty`

Bases: `pybind11_object`

`__eq__` (*self*: `rti.connextdds.WaitSetProperty`, *arg0*: `rti.connextdds.WaitSetProperty`) → bool

Compare objects for equality.

`__hash__` = None

`__init__` (**args*, ***kwargs*)

Overloaded function.

1. `__init__(self: rti.connextdds.WaitSetProperty) -> None`

Constructs a WaitSetProperty with default values.

2. `__init__(self: rti.connextdds.WaitSetProperty, event_count: int, event_delay: rti.connextdds.Duration) -> None`

Construct a WaitSetProperty with the given settings.

`__module__` = `'rti.connextdds'`

`__ne__` (*self*: `rti.connextdds.WaitSetProperty`, *arg0*: `rti.connextdds.WaitSetProperty`) → bool

Determine if objects are unequal.

property `event_count`

Get/set the event count that will wake the WaitSet.

property `event_delay`

Get/set the delay that will cause the WaitSet to wake even if the event count is not hit.

class `rti.connextdds.WcharSeq`

Bases: `pybind11_object`

`__add__` (*self*: `rti.connextdds.WcharSeq`, *arg0*: `rti.connextdds.WcharSeq`) → `rti.connextdds.WcharSeq`

`__bool__` (*self*: `rti.connextdds.WcharSeq`) → bool

Check whether the list is nonempty

`__contains__` (*self*: `rti.connextdds.WcharSeq`, *x*: *str*) → bool

Return true the container contains x

`__delitem__` (**args*, ***kwargs*)

Overloaded function.

1. `__delitem__(self: rti.connextdds.WcharSeq, arg0: int) -> None`

Delete the list elements at index *i*

2. `__delitem__(self: rti.connextdds.WcharSeq, arg0: slice) -> None`

Delete list elements using a slice object

`__eq__ (self: rti.connextdds.WcharSeq, arg0: rti.connextdds.WcharSeq) → bool`

`__getitem__ (*args, **kwargs)`

Overloaded function.

1. `__getitem__(self: rti.connextdds.WcharSeq, s: slice) -> rti.connextdds.WcharSeq`

Retrieve list elements using a slice object

2. `__getitem__(self: rti.connextdds.WcharSeq, arg0: int) -> str`

`__hash__ = None`

`__iadd__ (self: rti.connextdds.WcharSeq, arg0: rti.connextdds.WcharSeq) → rti.connextdds.WcharSeq`

`__imul__ (self: rti.connextdds.WcharSeq, arg0: int) → rti.connextdds.WcharSeq`

`__init__ (*args, **kwargs)`

Overloaded function.

1. `__init__(self: rti.connextdds.WcharSeq) -> None`
2. `__init__(self: rti.connextdds.WcharSeq, arg0: rti.connextdds.WcharSeq) -> None`

Copy constructor

3. `__init__(self: rti.connextdds.WcharSeq, arg0: Iterable) -> None`

`__iter__ (self: rti.connextdds.WcharSeq) → Iterator`

`__len__ (self: rti.connextdds.WcharSeq) → int`

`__module__ = 'rti.connextdds'`

`__mul__ (self: rti.connextdds.WcharSeq, arg0: int) → rti.connextdds.WcharSeq`

`__ne__ (self: rti.connextdds.WcharSeq, arg0: rti.connextdds.WcharSeq) → bool`

`__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>`

`__repr__ (self: rti.connextdds.WcharSeq) → str`

Return the canonical string representation of this list.

`__rmul__ (self: rti.connextdds.WcharSeq, arg0: int) → rti.connextdds.WcharSeq`

__setitem__ (*args, **kwargs)

Overloaded function.

1. `__setitem__(self: rti.connextdds.WcharSeq, arg0: int, arg1: str) -> None`
2. `__setitem__(self: rti.connextdds.WcharSeq, arg0: slice, arg1: rti.connextdds.WcharSeq) -> None`

Assign list elements using a slice object

append (self: rti.connextdds.WcharSeq, x: str) → None

Add an item to the end of the list

clear (self: rti.connextdds.WcharSeq) → None

Clear the contents

count (self: rti.connextdds.WcharSeq, x: str) → int

Return the number of times x appears in the list

extend (*args, **kwargs)

Overloaded function.

1. `extend(self: rti.connextdds.WcharSeq, L: rti.connextdds.WcharSeq) -> None`

Extend the list by appending all the items in the given list

2. `extend(self: rti.connextdds.WcharSeq, L: Iterable) -> None`

Extend the list by appending all the items in the given list

insert (self: rti.connextdds.WcharSeq, i: int, x: str) → None

Insert an item at a given position.

pop (*args, **kwargs)

Overloaded function.

1. `pop(self: rti.connextdds.WcharSeq) -> str`

Remove and return the last item

2. `pop(self: rti.connextdds.WcharSeq, i: int) -> str`

Remove and return the item at index i

remove (self: rti.connextdds.WcharSeq, x: str) → None

Remove the first item from the list whose value is x. It is an error if there is no such item.

class rti.connextdds.WcharType

Bases: *DynamicType*

__eq__ (self: rti.connextdds.WcharType, arg0: rti.connextdds.WcharType) → bool

Test for equality.

__hash__ = None

`__init__` (*self*: rti.connexdds.WcharType) → None

Get the singleton for WcharType

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.WcharType, *arg0*: rti.connexdds.WcharType) → bool

Test for inequality.

class rti.connexdds.WireProtocol

Bases: pybind11_object

`RTPS_AUTO_ID` = 0

`__eq__` (*self*: rti.connexdds.WireProtocol, *arg0*: rti.connexdds.WireProtocol) → bool

Test for equality.

`__hash__` = None

`__init__` (*self*: rti.connexdds.WireProtocol) → None

Creates the default policy.

`__module__` = 'rti.connexdds'

`__ne__` (*self*: rti.connexdds.WireProtocol, *arg0*: rti.connexdds.WireProtocol) → bool

Test for inequality.

property check_crc

Checks if the received RTPS message is valid by comparing the computed CRC with the received RTPS CRC submessage when this field is set to true.

property compute_crc

Adds RTPS CRC submessage to every message when this field is set to true.

property participant_id

A value used to distinguish among different participants belonging to the same domain on the same host.

property rtps_app_id

The RTPS App ID of the domain participant.

property rtps_auto_id_kind

Kind of auto mechanism used to calculate the GUID prefix.

property rtps_host_id

The RTPS Host ID of the domain participant.

property rtps_instance_id

The RTPS Instance ID of the domain participant.

property rtps_reserved_port_mask

Specifies which well-known ports to reserve when enabling the participant.

This property's getter returns a deep copy.

property rtps_well_known_ports

The RTPS well-known port mappings.

This property's getter returns a deep copy.

class rti.connextdds.WireProtocolAutoKind

Bases: pybind11_object

RTPS_AUTO_ID_FROM_IP = <WireProtocolAutoKind.RTPS_AUTO_ID_FROM_IP: 0>

RTPS_AUTO_ID_FROM_MAC =
<WireProtocolAutoKind.RTPS_AUTO_ID_FROM_MAC: 1>

RTPS_AUTO_ID_FROM_UUID =
<WireProtocolAutoKind.RTPS_AUTO_ID_FROM_UUID: 2>

class WireProtocolAutoKind

Bases: pybind11_object

Members:

RTPS_AUTO_ID_FROM_IP : Select the IPv4 based algorithm.

RTPS_AUTO_ID_FROM_MAC : Select the MAC based algorithm.

RTPS_AUTO_ID_FROM_UUID : Select the UUID based algorithm.

RTPS_AUTO_ID_FROM_IP =
<WireProtocolAutoKind.RTPS_AUTO_ID_FROM_IP: 0>

RTPS_AUTO_ID_FROM_MAC =
<WireProtocolAutoKind.RTPS_AUTO_ID_FROM_MAC: 1>

RTPS_AUTO_ID_FROM_UUID =
<WireProtocolAutoKind.RTPS_AUTO_ID_FROM_UUID: 2>

__eq__ (*self: object, other: object*) → bool

__getstate__ (*self: object*) → int

__hash__ (*self: object*) → int

__index__ (*self: rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind*) → int

__init__ (*self: rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind, value: int*) → None

__int__ (*self: rti.connextdds.WireProtocolAutoKind.WireProtocolAutoKind*) → int

```

__members__ = {'RTPS_AUTO_ID_FROM_IP':
<WireProtocolAutoKind.RTPS_AUTO_ID_FROM_IP: 0>,
'RTPS_AUTO_ID_FROM_MAC':
<WireProtocolAutoKind.RTPS_AUTO_ID_FROM_MAC: 1>,
'RTPS_AUTO_ID_FROM_UUID':
<WireProtocolAutoKind.RTPS_AUTO_ID_FROM_UUID: 2>}

```

```
__module__ = 'rti.connexdds'
```

```
__ne__ (self: object, other: object) → bool
```

```
__repr__ (self: object) → str
```

```
__setstate__ (self: rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind, state:
int) → None
```

```
__str__ ()
```

```
name(self: handle) -> str
```

property name

property value

```
__eq__ (self: rti.connexdds.WireProtocolAutoKind, arg0: rti.connexdds.WireProtocolAutoKind)
→ bool
```

Apply operator to underlying enumerated values.

```
__ge__ (self: rti.connexdds.WireProtocolAutoKind, arg0: rti.connexdds.WireProtocolAutoKind)
→ bool
```

Apply operator to underlying enumerated values.

```
__gt__ (self: rti.connexdds.WireProtocolAutoKind, arg0: rti.connexdds.WireProtocolAutoKind)
→ bool
```

Apply operator to underlying enumerated values.

```
__hash__ = None
```

```
__init__ (*args, **kwargs)
```

Overloaded function.

1. `__init__(self: rti.connexdds.WireProtocolAutoKind) -> None`

Initializes enum to 0.

2. `__init__(self: rti.connexdds.WireProtocolAutoKind, arg0: rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind) -> None`

Copy constructor.

```
__int__ (self: rti.connexdds.WireProtocolAutoKind) →
rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind
```

Int conversion.

__le__ (*self*: rti.connexdds.WireProtocolAutoKind, *arg0*: rti.connexdds.WireProtocolAutoKind) → bool

Apply operator to underlying enumerated values.

__lt__ (*self*: rti.connexdds.WireProtocolAutoKind, *arg0*: rti.connexdds.WireProtocolAutoKind) → bool

Apply operator to underlying enumerated values.

__module__ = 'rti.connexdds'

__ne__ (*self*: rti.connexdds.WireProtocolAutoKind, *arg0*: rti.connexdds.WireProtocolAutoKind) → bool

Apply operator to underlying enumerated values.

__str__ (*self*: rti.connexdds.WireProtocolAutoKind) → str

String conversion.

property underlying

Retrieves the actual enumerated value.

class rti.connexdds.**WriteParams**

Bases: pybind11_object

__init__ (*self*: rti.connexdds.WriteParams) → None

Create a WriteParams object with default values.

__module__ = 'rti.connexdds'

property cookie

The cookie for writing.

property flag

The sample flag for writing.

property handle

The instance handle for writing.

property identity

The sample identity.

property priority

The priority for writing.

property related_reader_guid

The related reader GUID for writing.

property related_sample_identity

The related sample identity.

property related_source_guid

The related source GUID for writing.

property replace_automatic_values

Indicates if the replacement of automatic values has been activated or not.

reset (*self*: rti.connextdds.WriteParams) → None

Reset all fields to their default values.

property source_guid

The source GUID for writing.

property source_timestamp

The source timestamp for writing.

class rti.connextdds.WriterDataLifecycle

Bases: pybind11_object

__eq__ (*self*: rti.connextdds.WriterDataLifecycle, *arg0*: rti.connextdds.WriterDataLifecycle) → bool

Test for equality.

__hash__ = None

__init__ (**args*, ***kwargs*)

Overloaded function.

1. **__init__**(*self*: rti.connextdds.WriterDataLifecycle) -> None

Creates the default policy.

2. **__init__**(*self*: rti.connextdds.WriterDataLifecycle, *autodispose*: bool) -> None

Creates an instance with a value for auto-dispose unregistered instances and default values for the rest of parameters.

__module__ = 'rti.connextdds'

__ne__ (*self*: rti.connextdds.WriterDataLifecycle, *arg0*: rti.connextdds.WriterDataLifecycle) → bool

Test for inequality.

property autodispose_unregistered_instances

Indicates whether the DataWriter should automatically dispose an instance when it unregisters it.

property autopurge_disposed_instances_delay

Maximum duration for which the DataWriter will maintain information regarding an instance once it has disposed the instance.

This property's getter returns a deep copy.

property autopurge_unregistered_instances_delay

Maximum duration for which the DataWriter will maintain information regarding an instance once it has unregistered the instance.

This property's getter returns a deep copy.

class rti.connextdds.WstringSeq

Bases: pybind11_object

__add__ (*self*: rti.connexdds.WstringSeq, *arg0*: rti.connexdds.WstringSeq) → *rti.connexdds.WstringSeq*

__bool__ (*self*: rti.connexdds.WstringSeq) → bool
Check whether the list is nonempty

__contains__ (*self*: rti.connexdds.WstringSeq, *x*: *str*) → bool
Return true the container contains *x*

__delitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__delitem__**(*self*: rti.connexdds.WstringSeq, *arg0*: int) -> None
Delete the list elements at index *i*
2. **__delitem__**(*self*: rti.connexdds.WstringSeq, *arg0*: slice) -> None
Delete list elements using a slice object

__eq__ (*self*: rti.connexdds.WstringSeq, *arg0*: rti.connexdds.WstringSeq) → bool

__getitem__ (**args*, ***kwargs*)
Overloaded function.

1. **__getitem__**(*self*: rti.connexdds.WstringSeq, *s*: slice) -> rti.connexdds.WstringSeq
Retrieve list elements using a slice object
2. **__getitem__**(*self*: rti.connexdds.WstringSeq, *arg0*: int) -> str

__hash__ = None

__iadd__ (*self*: rti.connexdds.WstringSeq, *arg0*: rti.connexdds.WstringSeq) → *rti.connexdds.WstringSeq*

__imul__ (*self*: rti.connexdds.WstringSeq, *arg0*: int) → *rti.connexdds.WstringSeq*

__init__ (**args*, ***kwargs*)
Overloaded function.

1. **__init__**(*self*: rti.connexdds.WstringSeq) -> None
2. **__init__**(*self*: rti.connexdds.WstringSeq, *arg0*: rti.connexdds.WstringSeq) -> None
Copy constructor
3. **__init__**(*self*: rti.connexdds.WstringSeq, *arg0*: Iterable) -> None

__iter__ (*self*: rti.connexdds.WstringSeq) → Iterator

__len__ (*self*: rti.connexdds.WstringSeq) → int

__module__ = 'rti.connexdds'

__mul__ (*self*: rti.connexdds.WstringSeq, *arg0*: int) → *rti.connexdds.WstringSeq*

__ne__ (*self*: rti.connexdds.WstringSeq, *arg0*: rti.connexdds.WstringSeq) → bool

__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__ = <capsule object NULL>

__rmul__ (*self*: rti.connexdds.WstringSeq, *arg0*: int) → rti.connexdds.WstringSeq

__setitem__ (**args*, ***kwargs*)

Overloaded function.

1. **__setitem__**(*self*: rti.connexdds.WstringSeq, *arg0*: int, *arg1*: str) -> None
2. **__setitem__**(*self*: rti.connexdds.WstringSeq, *arg0*: slice, *arg1*: rti.connexdds.WstringSeq) -> None

Assign list elements using a slice object

append (*self*: rti.connexdds.WstringSeq, *x*: str) → None

Add an item to the end of the list

clear (*self*: rti.connexdds.WstringSeq) → None

Clear the contents

count (*self*: rti.connexdds.WstringSeq, *x*: str) → int

Return the number of times *x* appears in the list

extend (**args*, ***kwargs*)

Overloaded function.

1. **extend**(*self*: rti.connexdds.WstringSeq, *L*: rti.connexdds.WstringSeq) -> None

Extend the list by appending all the items in the given list

2. **extend**(*self*: rti.connexdds.WstringSeq, *L*: Iterable) -> None

Extend the list by appending all the items in the given list

insert (*self*: rti.connexdds.WstringSeq, *i*: int, *x*: str) → None

Insert an item at a given position.

pop (**args*, ***kwargs*)

Overloaded function.

1. **pop**(*self*: rti.connexdds.WstringSeq) -> str

Remove and return the last item

2. **pop**(*self*: rti.connexdds.WstringSeq, *i*: int) -> str

Remove and return the item at index *i*

remove (*self*: rti.connexdds.WstringSeq, *x*: str) → None

Remove the first item from the list whose value is *x*. It is an error if there is no such item.

`rti.connexdds.set_activity_context` (*attribute_mask*: `rti.connexdds.ActivityContextMask`)
→ None

Set the activity context mask.

`rti.connexdds.sleep` (*duration*: `rti.connexdds.Duration`) → None

Sleep for a specified duration

`rti.connexdds.spin` (*spin_count*: `int`) → None

Performs a spin operation (active wait) as many times as indicated.

`rti.connexdds.spin_per_microsecond`() → `int`

Returns the number of spin operations needed to wait 1 microsecond

`rti.connexdds.unregister_thread`() → None

Unregister a thread with the middleware

2.3.2 rti.types

This module defines the `struct`, `union`, `enum`, `alias` decorators that allow turning Python data-classes into DDS topic-types.

See *Data Types* for more information.

`rti.types` also contains `rti.types.builtin`.

rti.types.builtin

This submodule defines the DDS built-in types. These simple, general-purpose types are available for the user's convenience to quickly write a publisher or subscriber application.

class `rti.types.builtin.Bytes` (*value*: `~typing.Sequence[uint8]`) = `<factory>`

Built-in type consisting of a sequence of bytes

This type can be used to create a `dds.Topic`, e.g. `topic = dds.Topic(participant, "My Topic", Bytes)`.

value: Sequence[uint8]

The byte payload

class `rti.types.builtin.KeyedBytes` (*key*: `str = ""`, *value*: `~typing.Sequence[uint8]`) = `<factory>`

Built-in type consisting of a sequence of bytes and a string that is the key.

This type can be used to create a `dds.Topic`, e.g. `topic = dds.Topic(participant, "My Topic", KeyedBytes)`.

key: str = ''

The key

value: Sequence[uint8]

The byte payload

class `rti.types.builtin.KeyedString` (*key: str = "", value: str = ""*)

Built-in type consisting of a string payload and a second string that is the key.

This type can be used to create a `dds.Topic`, e.g. `topic = dds.Topic(participant, "MyTopic", KeyedString)`.

key: str = ''

The key

value: str = ''

The string payload

class `rti.types.builtin.String` (*value: str = ""*)

Built-in type consisting of a string

This type can be used to create a `dds.Topic`, e.g. `topic = dds.Topic(participant, "MyTopic", String)`

value: str = ''

The string payload

2.3.3 rti.rpc

`rti.rpc` is the package containing the RTI Connext Request-Reply and Remote Procedure APIs.

See *Request-Reply and Remote Procedure Calls* for an overview of the API.

class `rti.rpc.Requester` (*request_type: Union[type, DynamicType], reply_type: Union[type, DynamicType], participant: DomainParticipant, service_name: Optional[str] = None, request_topic: Optional[Union[Topic, Topic, str, object]] = None, reply_topic: Optional[Union[Topic, Topic, str, object]] = None, datawriter_qos: Optional[DataWriterQos] = None, datareader_qos: Optional[DataReaderQos] = None, publisher: Optional[Publisher] = None, subscriber: Optional[Subscriber] = None, on_reply_available: Optional[Callable[[object], object]] = None*)

A Requester allows sending requests and receiving replies

Parameters

- **request_type** – The type of the request data. It can be an `@idl.struct`, an `@idl.union`, or a `dds.DynamicType`. (See *Data Types*.)
- **reply_type** – The type of the reply data.
- **participant** – The `DomainParticipant` that will hold the request writer and reply reader.
- **service_name** – Name that will be used to derive the topic name, defaults to `None` (rely only on custom topics).

- **request_topic** – Topic object or name that will be used for the request data, must be set if `service_name` is `None`, otherwise overrides `service_name`, defaults to `None` (use `service_name`).
- **reply_topic** – Topic object or name that will be used for the reply data, must be set if `service_name` is `None`, otherwise overrides `service_name`, defaults to `None` (use `service_name`).
- **datawriter_qos** – QoS object to use for request writer, defaults to `None` (use default `RequestReply` QoS).
- **datareader_qos** – QoS object to use for reply reader, defaults to `None` (use default `RequestReply` QoS).
- **publisher** – Publisher used to hold request writer, defaults to `None` (use participant builtin publisher).
- **subscriber** – Subscriber used to hold reply reader, defaults to `None` (use participant builtin subscriber).
- **on_reply_available** – The callback that handles incoming replies.

__init__ (*request_type: Union[type, DynamicType], reply_type: Union[type, DynamicType], participant: DomainParticipant, service_name: Optional[str] = None, request_topic: Optional[Union[Topic, Topic, str, object]] = None, reply_topic: Optional[Union[Topic, Topic, str, object]] = None, datawriter_qos: Optional[DataWriterQos] = None, datareader_qos: Optional[DataReaderQos] = None, publisher: Optional[Publisher] = None, subscriber: Optional[Subscriber] = None, on_reply_available: Optional[Callable[[object], object]] = None*) → `None`

__weakref__

list of weak references to the object (if defined)

close () → `None`

Close the resources for this request-reply object.

property closed: bool

Returns true if this request-reply object has been closed.

Getter

Returns the number of matched requesters.

classmethod is_final_reply (*reply_info: Union[SampleInfo, object]*) → `bool`

Check a reply is the last of the sequence.

Parameters

reply_info – The reply info with the flags to check.

Returns

Boolean indicating whether reply is the last for a request.

classmethod is_related_reply (*request_id: SampleIdentity, reply_info: SampleInfo*) → `bool`

Check a request if against a reply's metadata for correlation.

Parameters

- **request_id** – The request id used to correlate replies.
- **reply_info** – The reply info used for the correlation check.

Returns

Boolean indicating whether the request and reply are correlated.

property matched_replier_count: int

The number of discovered matched repliers.

Getter

Returns the number of matched repliers.

property on_reply_available: Optional[Callable[[object], object]]

The listener callback used to process received replies.

Getter

Returns the callback function.

Setter

Set the callback function.

read_replies (*related_request_id: Optional[SampleIdentity] = None*) → Union[*LoanedSamples*,
LoanedSamples]

Read received replies.

Parameters

related_request_id – The id used to correlate replies to a specific request, default None (read any replies).

Returns

A loaned samples object containing the replies.

receive_replies (*max_wait: Duration, min_count: int = 1, related_request_id: Optional[SampleIdentity] = None*) → Union[*LoanedSamples*,
LoanedSamples, object]

Wait for replies and take them.

Parameters

- **max_wait** – Maximum time to wait for replies before timing out.
- **min_count** – Minimum number of replies to receive, default 1.
- **related_request_id** – The request id used to correlate replies, default None (receive any replies).

Raises

dds.TimeoutError – Thrown if min_count not received within max_wait.

Returns

A loaned samples object containing the replies.

property reply_datareader: Union[DataReader, DataReader, object]

The DataReader used to receive reply data.

Getter

Returns the reply DataReader.

property request_datawriter: Union[DataWriter, DataWriter]

The DataWriter used to send request data.

Getter

Returns the request DataWriter.

send_request (*request: Union[object, DynamicData], params: Optional[WriteParams] = None*)
→ SampleIdentity

Send a request and return the identity of the request for correlating received replies.

Parameters

- **request** – The request to send.
- **params** – Parameters used for writing the request.

Returns

The identity of the request.

take_replies (*related_request_id: Optional[SampleIdentity] = None*) → Union[LoanedSamples, LoanedSamples]

Take received replies.

Parameters

related_request_id – The id used to correlate replies to a specific request, default None (take any replies).

Returns

A loaned samples object containing the replies.

wait_for_replies (*max_wait: Duration, min_count: int = 1, related_request_id: Optional[SampleIdentity] = None*) → bool

Wait for received replies.

Parameters

- **max_wait** – Maximum time to wait for replies before timing out.
- **min_count** – Minimum number of replies to receive, default 1.
- **related_request_id** – The request id used to correlate replies, default None (receive any replies).

Returns

Boolean indicating whether min_count replies were received within max_wait time.

async wait_for_replies_async (*max_wait: Duration, min_count: int = 1, related_request_id: Optional[SampleIdentity] = None*) → bool

Wait for received replies asynchronously.

Parameters

- **max_wait** – Maximum time to wait for replies before timing out.
- **min_count** – Minimum number of replies to receive, default 1.
- **related_request_id** – The request id used to correlate replies, default None (receive any replies).

Returns

Boolean indicating whether min_count replies were received within max_wait time.

```
class rti.rpc.Replier (request_type: Union[type, DynamicType], reply_type: Union[type, DynamicType], participant: DomainParticipant, service_name: Optional[str] = None, request_topic: Optional[Union[Topic, ContentFilteredTopic, str, object]] = None, reply_topic: Optional[Union[Topic, str, object]] = None, datawriter_qos: Optional[DataWriterQos] = None, datareader_qos: Optional[DataReaderQos] = None, publisher: Optional[Publisher] = None, subscriber: Optional[Subscriber] = None, on_request_available: Optional[Callable[[object], object]] = None)
```

A replier object for handling request-reply interactions with DDS.

Parameters

- **request_type** – The type of the request data.
- **reply_type** – The type of the reply data.
- **participant** – The DomainParticipant that will hold the reply writer and request reader.
- **service_name** – Name that will be used to derive the topic name, defaults to None (rely only on custom topics).
- **request_topic** – Topic object or name that will be used for the request data, must be set if service_name is None, otherwise overrides service_name, defaults to None (use service_name).
- **reply_topic** – Topic object or name that will be used for the reply data, must be set if service_name is None, otherwise overrides service_name, defaults to None (use service_name).
- **datawriter_qos** – QoS object to use for reply writer, defaults to None (use default RequestReply QoS).
- **datareader_qos** – QoS object to use for request reader, defaults to None (use default RequestReply QoS).
- **publisher** – Publisher used to hold reply writer, defaults to None (use participant builtin publisher).

- **subscriber** – Subscriber used to hold request reader, defaults to None (use participant builtin subscriber).
- **on_reply_available** – The callback that handles incoming requests.

__init__ (*request_type: Union[type, DynamicType], reply_type: Union[type, DynamicType], participant: DomainParticipant, service_name: Optional[str] = None, request_topic: Optional[Union[Topic, ContentFilteredTopic, str, object]] = None, reply_topic: Optional[Union[Topic, str, object]] = None, datawriter_qos: Optional[DataWriterQos] = None, datareader_qos: Optional[DataReaderQos] = None, publisher: Optional[Publisher] = None, subscriber: Optional[Subscriber] = None, on_request_available: Optional[Callable[[object], object]] = None*) → None

__weakref__

list of weak references to the object (if defined)

close () → None

Close the resources for this request-reply object.

property closed: bool

Returns true if this request-reply object has been closed.

Getter

Returns the number of matched requesters.

property matched_requester_count: int

The number of discovered matched requesters.

Getter

Returns the number of matched requesters.

property on_request_available

The listener callback used to process received requests.

Getter

Returns the callback function.

Setter

Set the callback function.

Type

Optional[Callable[[*Replier*]]]

read_requests () → Union[*LoanedSamples*, *LoanedSamples*]

Read received requests.

Returns

A loaned samples object containing the requests.

receive_requests (*max_wait: Duration, min_count: int = 1*) → Union[*LoanedSamples*, *LoanedSamples*]

Receive a minimum number of requests within a timeout period.

Parameters

- **max_wait** – Maximum time to wait for requests before timing out. .
- **min_count** – Minimum number of requests to receive, default 1.

Raises

dds.TimeoutError – Thrown if min_count not received within max_wait.

Returns

A loaned samples object containing the requests.

property reply_datawriter: Union[DataWriter, DataWriter]

The DataWriter used to send reply data.

Getter

Returns the reply DataWriter.

property request_datareader: Union[DataReader, DataReader]

The DataReader used to receive request data.

Getter

Returns the request DataReader.

send_reply (*reply: Union[DynamicData, object]*, *param: Union[SampleIdentity, SampleInfo, WriteParams]*, *final: bool = True*) → None

Send a reply to a received request.

Parameters

- **reply** – The reply to send.
- **param** – Parameters used for writing the request.
- **final** – Indicates whether this is the final reply for a specific request, default True.

Raises

dds.InvalidArgumentError – Thrown if param is not a type that can be used for correlation.

take_requests () → Union[LoanedSamples, LoanedSamples]

Take received requests.

Returns

A loaned samples object containing the requests.

Return type

Union[dds.DynamicData.LoanedSamples, object]

wait_for_requests (*max_wait: Duration*, *min_count: int = 1*) → bool

Wait for a minimum number of requests within a timeout period.

Parameters

- **max_wait** – Maximum time to wait for requests before timing out. .
- **min_count** – Minimum number of requests to receive, default 1.

Returns

Boolean indicating whether `min_count` requests were received within `max_wait` time.

async wait_for_requests_async (*max_wait*: Duration, *min_count*: Optional[int] = 1) → bool

Wait asynchronously for a minimum number of requests within a timeout period.

Parameters

- **max_wait** – Maximum time to wait for requests before timing out. .
- **min_count** – Minimum number of requests to receive, default 1.

Returns

Boolean indicating whether `min_count` requests were received within `max_wait` time.

```
class rti.rpc.SimpleReplier (request_type: Union[DynamicType, type], reply_type:
    Union[DynamicType, type], participant: DomainParticipant,
    handler: Callable[[object], object], service_name: Optional[str] =
    None, request_topic: Optional[Union[Topic, ContentFilteredTopic,
    str, object]] = None, reply_topic: Optional[Union[Topic, str,
    object]] = None, datawriter_qos: Optional[DataWriterQos] =
    None, datareader_qos: Optional[DataReaderQos] = None,
    publisher: Optional[Publisher] = None, subscriber:
    Optional[Subscriber] = None)
```

A special replier that uses a user callback to produce one reply per request.

Parameters

- **request_type** – The type of the request data.
- **reply_type** – The type of the reply data.
- **participant** – The DomainParticipant that will hold the request reader and reply writer.
- **handler** – The callback that handles incoming requests and returns a reply. The callback must have a single argument of type `request_type` and must return an instance of type `reply_type`.
- **service_name** – Name that will be used to derive the topic name, defaults to None (rely only on custom topics).
- **request_topic** – Topic object or name that will be used for the request data, must be set if `service_name` is None, otherwise overrides `service_name`, defaults to None (use `service_name`).
- **reply_topic** – Topic object or name that will be used for the reply data, must be set if `service_name` is None, otherwise overrides `service_name`, defaults to None (use `service_name`).
- **datawriter_qos** – QoS object to use for reply writer, defaults to None (use default RequestReply QoS).

- **datareader_qos** – QoS object to use for request reader, defaults to None (use default RequestReply QoS).
- **publisher** – Publisher used to hold reply writer, defaults to None (use participant builtin publisher).
- **subscriber** – Subscriber used to hold request reader, defaults to None (use participant builtin subscriber).

__init__ (*request_type: Union[DynamicType, type], reply_type: Union[DynamicType, type], participant: DomainParticipant, handler: Callable[[object], object], service_name: Optional[str] = None, request_topic: Optional[Union[Topic, ContentFilteredTopic, str, object]] = None, reply_topic: Optional[Union[Topic, str, object]] = None, datawriter_qos: Optional[DataWriterQos] = None, datareader_qos: Optional[DataReaderQos] = None, publisher: Optional[Publisher] = None, subscriber: Optional[Subscriber] = None*) → None

__weakref__

list of weak references to the object (if defined)

close () → None

Close the resources for this request-reply object.

property closed: bool

Returns true if this request-reply object has been closed.

Getter

Returns the number of matched requesters.

property matched_requester_count: int

The number of discovered matched requesters.

Getter

Returns the number of matched requesters.

class `rti.rpc.Service` (*service_instance: ABC, participant: DomainParticipant, service_name: str, task_count: int = 4, datawriter_qos: Optional[DataWriterQos] = None, datareader_qos: Optional[DataReaderQos] = None, publisher: Optional[Publisher] = None, subscriber: Optional[Subscriber] = None*)

A service allows running a `service_instance` in a DDS domain using `asyncio`.

The service uses a Replier to receive RPC calls and then dispatches them to the `service_instance`, calling the appropriate method. The value returned by the method is then sent back to the remote caller.

The service runs asynchronously (run method) until the task is cancelled.

close ()

Closes the DDS entities used by this service.

property matched_client_count: int

The number of RPC clients that match this service.

async run (*close_on_cancel: bool = False*)

Starts receiving RPC calls (requests) and processing them.

This method runs until the task it returns is cancelled.

If *close_on_cancel* is True, the service will close the DDS entities when the task is canceled. By default it is False, which means you can call `run()` again after a `run()` task is cancelled.

Exceptions raised during the execution of the service are logged as warnings module and do not stop the execution of the service.

```
class rti.rpc.ClientBase (participant: ~rti.connexdds.DomainParticipant, service_name: str,
                        max_wait_per_call: ~rti.connexdds.Duration =
                        <rti.connexdds.Duration object>, datawriter_qos:
                        ~typing.Optional[~rti.connexdds.DataWriterQos] = None,
                        datareader_qos: ~typing.Optional[~rti.connexdds.DataReaderQos] =
                        None, publisher: ~typing.Optional[~rti.connexdds.Publisher] = None,
                        subscriber: ~typing.Optional[~rti.connexdds.Subscriber] = None)
```

Base class for RPC clients.

An actual Client must inherit from a service interface and from this class, for example:

```
` class RobotClient(Robot, rpc.ClientBase): ... `
```

This base class injects an implementation for all the `@operation` methods found in `Robot`, which uses a `Requester` to make RPC calls and return the values it receives.

The base class also provides an `__init__`, `close` and other methods.

close ()

Closes the DDS entities used by this client.

property matched_service_count: int

The number of RPC services that match this client.

```
@rti.rpc.service (cls=None, *, type_annotations=[], member_annotations={})
```

This decorator marks an abstract base class as a remote service interface.

A class annotated with this decorator can be used to create a `Client` or to define the implementation to be run in a `Service`.

The operations that will be remotely callable need to be marked with the `@operation` decorator.

```
@rti.rpc.operation (funcobj=None, *, raises=[], parameter_annotations={})
```

This decorator marks a method as an remote operation of a service interface.

It also marks it as an `@abc.abstractmethod`.

Only methods marked with this decorator will be callable using an `RPC Client` or an `RPC Service`.

exception rti.rpc.RemoteUnknownOperationError

Exception thrown by a client operation when the server indicates that the operation is unknown to the server.

exception `rti.rpc.RemoteUnknownExceptionError`

Exception thrown by a client operation when the server operation fails with an exception that is not declared in the interface.

2.3.4 rti.asyncio

Note: This module requires Python 3.7 or newer

This module must be imported in order to use the methods `rti.connextdds.DataReader.take_async()` and `rti.connextdds.DataReader.take_data_async()`.

These two methods are added to the `DataReader` class when this module is imported.

The module also defines a convenience function `rti.asyncio.run`, which is similar to `asyncio.run`, and can synchronously run the main async function in an application.

See *Subscriptions*.

`rti.asyncio.run` (*coroutine*)

Uses the current event loop to run the given coroutine and waits until it finishes. If there is no current running event loop, a new one is created

2.3.5 rti.logging

Submodules

rti.logging.distlog module

class `rti.logging.distlog.LogLevel`

Bases: `pybind11_object`

Members:

SILENT

FATAL

SEVERE

ERROR

WARNING

NOTICE

INFO

DEBUG

TRACE

```

DEBUG = <LogLevel.DEBUG: 700>
ERROR = <LogLevel.ERROR: 300>
FATAL = <LogLevel.FATAL: 100>
INFO = <LogLevel.INFO: 600>
NOTICE = <LogLevel.NOTICE: 500>
SEVERE = <LogLevel.SEVERE: 200>
SILENT = <LogLevel.SILENT: 0>
TRACE = <LogLevel.TRACE: 800>
WARNING = <LogLevel.WARNING: 400>

```

property name

property value

```
class rti.logging.distlog.Logger
```

Bases: pybind11_object

```
static debug(message: str) → None
```

Log a debug message.

```
static error(message: str) → None
```

Log an error message.

```
static fatal(message: str) → None
```

Log a fatal message.

```
static filter_level(level: rti.logging.distlog.LogLevel) → None
```

The logger filter level.

```
static finalize() → None
```

Destroy the Logger. It should not be accessed after this call.

```
static info(message: str) → None
```

Log an info message.

```
static init(options: Optional[rti.logging.distlog.LoggerOptions] = None) → None
```

Initializes the distributed logger

```
static log(*args, **kwargs)
```

Overloaded function.

1. log(log_level: rti.logging.distlog.LogLevel, message: str) -> None

Log a message with the given log level.

2. log(log_level: rti.logging.distlog.LogLevel, message: str, category: str) -> None

Log a message with the given log level and category.

3. `log(message_params: rti.logging.distlog.MessageParams) -> None`

Log a message with the given message parameters.

static notice (*message: str*) → None

Log a notice message.

static print_format (*format: rti.connextdds.PrintFormat*) → None

The logger print format. NOTE: This will affect the print format of the associatedDomainParticipant's logger as well.

static severe (*message: str*) → None

Log a severe message.

static trace (*message: str*) → None

Log a trace message.

static verbosity (*category: rti.connextdds.LogCategory, level: rti.connextdds.Verbosity*) → None

The logger's verbosity. NOTE: This will affect the verbosity of the associatedDomainParticipant's logger as well.

static warning (*message: str*) → None

Log a warning message.

class `rti.logging.distlog.LoggerOptions`

Bases: `pybind11_object`

property application_kind

The application_kind.

property domain_id

The domain ID for logging.

property echo_to_stdout

Toggle for echo to stdout.

property filter_level

Toggle for log filter level.

property log_infrastructure_messages

Toggle for logging infrastructure messages.

property participant

The DomainParticipant to use for the logger.

property qos_library

The QoS library name.

property qos_profile

The QoS profile name.

property queue_size

The logger's queue size.

property remote_administration_enabled

Toggle for remote administration.

property thread_settings

The settings for the thread handling logging.

class rti.logging.distlog.MessageParams

Bases: pybind11_object

property category

The log message category.

property log_level

The message log level.

property message

The log message.

property timestamp

The timestamp of the log message.

rti.logging.handler module**class** rti.logging.handler.DistlogHandler (*options: Optional[LoggerOptions] = None*)

Bases: Handler

close () → None

Tidy up any resources used by the handler.

This version removes the handler from an internal map of handlers, `_handlers`, which is used for handler lookup by name. Subclasses should ensure that this gets called from overridden `close()` methods.

emit (*record: LogRecord*) → None

Do whatever it takes to actually log the specified logging record.

This version is intended to be implemented by subclasses and so raises a `NotImplementedError`.

Module contents

- *rti.connextdds* - the main package, containing the DDS API.
- *rti.types* - allows defining user types to be published and subscribed to.
- *rti.rpc* - contains the and Request-Reply and Remote Procedure APIs.
- *rti.asyncio* - adds support for asynchronous operations.
- *rti.logging* - provides logging utilities.

2.4 Examples

In addition to the code snippets in the [RTI Connex Getting Started Guide](#) and the *API Overview*, you can find full example applications in the [Connex Examples repository](#), including the following:

- [Content-Filtered Topic](#)
- [Content-Filtered Topic String Filter](#)
- [Built-in Topics](#)
- [Using Qos Profiles](#)
- [Partitions](#)
- [Group Coherent Presentation](#)
- [WaitSets and Query Conditions](#)
- [Asynchronous Publication](#)
- [Request-Reply](#)
- [Remote Procedure Call](#)
- [Basic Security](#)

2.5 Copyrights, Notices, License

© 2024 Real-Time Innovations, Inc. All rights reserved. Printed in U.S.A. First printing. February 2024.

Trademarks

RTI, Real-Time Innovations, Connex, NDDS, the RTI logo, 1RTI and the phrase, “Your Systems. Working as one.” are registered trademarks, trademarks or service marks of Real-Time Innovations, Inc. All other trademarks belong to their respective owners.

Copy and Use Restrictions

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form (including electronic, mechanical, photocopy, and facsimile) without the prior written permission of Real-Time Innovations, Inc. The software described in this document is furnished solely under and subject to RTI’s standard terms and conditions available at <https://www.rti.com/terms> and in accordance with your License Acknowledgement Certificate (LAC) and Maintenance and Support Certificate (MSC), except to the extent otherwise accepted in writing by a corporate officer of RTI.

This is an independent publication and is neither affiliated with, nor authorized, sponsored, or approved by, Microsoft Corporation.

The security features of this product include software developed by the OpenSSL Project for use in the OpenSSL Toolkit (<http://www.openssl.org/>). This product includes cryptographic software written by Eric Young (eay@cryptsoft.com). This product includes software written by Tim Hudson (tjh@cryptsoft.com).

Notices

Deprecations and Removals

Any deprecations or removals noted in this document serve as notice under the Real-Time Innovations, Inc. Maintenance Policy #4220 and/or any other agreements by and between RTI and customer regarding maintenance and support of RTI's software.

Deprecated means that the item is still supported in the release, but will be removed in a future release. *Removed* means that the item is discontinued or no longer supported. By specifying that an item is deprecated in a release, RTI hereby provides customer notice that RTI reserves the right after one year from the date of such release and, with or without further notice, to immediately terminate maintenance (including without limitation, providing updates and upgrades) for the item, and no longer support the item, in a future release.

Technical Support Real-Time Innovations, Inc. 232 E. Java Drive Sunnyvale, CA 94089 Phone: (408) 990-7444 Email: support@rti.com Website: <https://support.rti.com/>

Chapter 3

Additional documentation

For more documentation, including the Connex User's Manual and the reference for the C, C++, Java and C# APIs, see [RTI Community](#).

Python Module Index

r

- `rti.asyncio`, 770
- `rti.connextdds`, 31
- `rti.logging`, 773
- `rti.logging.distlog`, 770
- `rti.logging.handler`, 773
- `rti.types.builtin`, 759

Index

Non-alphabetical

- `__add__` () (*rti.connexdds.AnyDataReaderSeq* method), 43
- `__add__` () (*rti.connexdds.AnyDataWriterSeq* method), 46
- `__add__` () (*rti.connexdds.AnyTopicSeq* method), 49
- `__add__` () (*rti.connexdds.BoolSeq* method), 54
- `__add__` () (*rti.connexdds.ChannelSettingsSeq* method), 63
- `__add__` () (*rti.connexdds.CharSeq* method), 65
- `__add__` () (*rti.connexdds.ConditionSeq* method), 73
- `__add__` () (*rti.connexdds.ContentFilteredTopicSeq* method), 77
- `__add__` () (*rti.connexdds.CookieSeq* method), 80
- `__add__` () (*rti.connexdds.DataReaderSeq* method), 107
- `__add__` () (*rti.connexdds.DataWriterSeq* method), 138
- `__add__` () (*rti.connexdds.DomainParticipantSeq* method), 176
- `__add__` () (*rti.connexdds.Duration* method), 182
- `__add__` () (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 186
- `__add__` () (*rti.connexdds.DynamicData.DataReaderSeq* method), 195
- `__add__` () (*rti.connexdds.DynamicData.DataWriterSeq* method), 206
- `__add__` () (*rti.connexdds.DynamicDataSeq* method), 249
- `__add__` () (*rti.connexdds.DynamicDataTimestampedSeq* method), 251
- `__add__` () (*rti.connexdds.DynamicData.TopicSeq* method), 216
- `__add__` () (*rti.connexdds.EndpointGroupSeq* method), 256
- `__add__` () (*rti.connexdds.EntitySeq* method), 261
- `__add__` () (*rti.connexdds.EnumMemberSeq* method), 263
- `__add__` () (*rti.connexdds.Float32Seq* method), 274
- `__add__` () (*rti.connexdds.Float64Seq* method), 277
- `__add__` () (*rti.connexdds.Float128Seq* method), 272
- `__add__` () (*rti.connexdds.IAnyDataReaderSeq* method), 291
- `__add__` () (*rti.connexdds.IAnyDataWriterSeq* method), 294
- `__add__` () (*rti.connexdds.IAnyTopicSeq* method), 297
- `__add__` () (*rti.connexdds.IConditionSeq* method), 299
- `__add__` () (*rti.connexdds.IEntitySeq* method), 302
- `__add__` () (*rti.connexdds.InstanceHandleSeq* method), 309
- `__add__` () (*rti.connexdds.Int8Seq* method), 323
- `__add__` () (*rti.connexdds.Int16Seq* method), 314
- `__add__` () (*rti.connexdds.Int32Seq* method), 317
- `__add__` () (*rti.connexdds.Int64Seq* method), 320
- `__add__` () (*rti.connexdds.LocatorFilterElementSeq* method), 333
- `__add__` () (*rti.connexdds.LocatorSeq* method), 338
- `__add__` () (*rti.connexdds.MemberSeq* method), 347
- `__add__` () (*rti.connexdds.MonitoringMetricSelectionSeq* method), 353
- `__add__` () (*rti.connexdds.MulticastMappingSeq* method), 358
- `__add__` () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 371
- `__add__` () (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 381
- `__add__` () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 392
- `__add__` () (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 406
- `__add__` () (*rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq* method), 409
- `__add__` () (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 401
- `__add__` () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 427
- `__add__` () (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 437
- `__add__` () (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 449
- `__add__` () (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 464
- `__add__` () (*rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq* method), 466
- `__add__` () (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 458
- `__add__` () (*rti.connexdds.PublisherSeq* method), 476
- `__add__` () (*rti.connexdds.QosPolicyCountSeq* method), 478
- `__add__` () (*rti.connexdds.SequenceNumber* method), 519
- `__add__` () (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq* method), 525
- `__add__` () (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 534
- `__add__` () (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 545
- `__add__` () (*rti.connexdds.ServiceRequestSeq* method), 560
- `__add__` () (*rti.connexdds.ServiceRequestTimestampedSeq* method), 560

- method), 562
- __add__() (rti.connexdds.ServiceRequest.TopicSeq method), 553
- __add__() (rti.connexdds.StringPairSeq method), 572
- __add__() (rti.connexdds.StringSeq method), 574
- __add__() (rti.connexdds.SubscriberSeq method), 582
- __add__() (rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method), 586
- __add__() (rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq method), 595
- __add__() (rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq method), 607
- __add__() (rti.connexdds.SubscriptionBuiltinTopicDataSeq method), 622
- __add__() (rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq method), 625
- __add__() (rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq method), 616
- __add__() (rti.connexdds.Time method), 636
- __add__() (rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method), 641
- __add__() (rti.connexdds.TopicBuiltinTopicData.DataReaderSeq method), 651
- __add__() (rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method), 662
- __add__() (rti.connexdds.TopicBuiltinTopicDataSeq method), 676
- __add__() (rti.connexdds.TopicBuiltinTopicDataTimestampedSeq method), 678
- __add__() (rti.connexdds.TopicBuiltinTopicData.TopicSeq method), 671
- __add__() (rti.connexdds.TopicSeq method), 690
- __add__() (rti.connexdds.TransportInfoSeq method), 699
- __add__() (rti.connexdds.TransportMulticastSeq method), 706
- __add__() (rti.connexdds.TransportMulticastSettingsSeq method), 709
- __add__() (rti.connexdds.TransportUnicastSettingsSeq method), 713
- __add__() (rti.connexdds.Uint8Seq method), 734
- __add__() (rti.connexdds.Uint16Seq method), 726
- __add__() (rti.connexdds.Uint32Seq method), 729
- __add__() (rti.connexdds.Uint64Seq method), 731
- __add__() (rti.connexdds.UnionMemberSeq method), 738
- __add__() (rti.connexdds.WcharSeq method), 749
- __add__() (rti.connexdds.WstringSeq method), 756
- __and__() (rti.connexdds.ActivityContextMask method), 38
- __and__() (rti.connexdds.CompressionIdMask method), 70
- __and__() (rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method), 151
- __and__() (rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method), 154
- __and__() (rti.connexdds.InstanceState method), 311
- __and__() (rti.connexdds.RtpsReservedPortKindMask method), 501
- __and__() (rti.connexdds.SampleFlag method), 505
- __and__() (rti.connexdds.SampleLostState method), 510
- __and__() (rti.connexdds.SampleRejectedState method), 513
- __and__() (rti.connexdds.SampleState method), 516
- __and__() (rti.connexdds.StatusMask method), 567
- __and__() (rti.connexdds.StreamKind method), 569
- __and__() (rti.connexdds.ThreadSettingsKindMask method), 633
- __and__() (rti.connexdds.TransportBuiltinMask method), 693
- __and__() (rti.connexdds.ViewState method), 744
- __bool__() (rti.connexdds.ActivityContextMask method), 38
- __bool__() (rti.connexdds.AnyDataReaderSeq method), 43
- __bool__() (rti.connexdds.AnyDataWriterSeq method), 46
- __bool__() (rti.connexdds.AnyTopicSeq method), 49
- __bool__() (rti.connexdds.BoolSeq method), 54
- __bool__() (rti.connexdds.ChannelSettingsSeq method), 63
- __bool__() (rti.connexdds.CharSeq method), 65
- __bool__() (rti.connexdds.CompressionIdMask method), 70
- __bool__() (rti.connexdds.ConditionSeq method), 74
- __bool__() (rti.connexdds.ContentFilteredTopicSeq method), 77
- __bool__() (rti.connexdds.CookieSeq method), 80
- __bool__() (rti.connexdds.DataReaderSeq method), 107
- __bool__() (rti.connexdds.DataWriterSeq method), 138
- __bool__() (rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method), 151
- __bool__() (rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method), 154
- __bool__() (rti.connexdds.DomainParticipantSeq method), 176
- __bool__() (rti.connexdds.DynamicData.ContentFilteredTopicSeq method), 186
- __bool__() (rti.connexdds.DynamicData.DataReaderSeq method), 195
- __bool__() (rti.connexdds.DynamicData.DataWriterSeq method), 206
- __bool__() (rti.connexdds.DynamicDataSeq method), 249
- __bool__() (rti.connexdds.DynamicDataTimestampedSeq method), 252
- __bool__() (rti.connexdds.DynamicData.TopicSeq method), 216
- __bool__() (rti.connexdds.EndpointGroupSeq method), 256
- __bool__() (rti.connexdds.EntitySeq method), 261
- __bool__() (rti.connexdds.EnumMemberSeq method), 264
- __bool__() (rti.connexdds.Float32Seq method), 274
- __bool__() (rti.connexdds.Float64Seq method), 277
- __bool__() (rti.connexdds.Float128Seq method), 272
- __bool__() (rti.connexdds.IAnyDataReaderSeq method), 291
- __bool__() (rti.connexdds.IAnyDataWriterSeq method), 294

- `__bool__` () (*rti.connexdds.IAnyTopicSeq* method), 297
- `__bool__` () (*rti.connexdds.IConditionSeq* method), 299
- `__bool__` () (*rti.connexdds.IEntitySeq* method), 302
- `__bool__` () (*rti.connexdds.InstanceHandle* method), 308
- `__bool__` () (*rti.connexdds.InstanceHandleSeq* method), 309
- `__bool__` () (*rti.connexdds.InstanceState* method), 311
- `__bool__` () (*rti.connexdds.Int8Seq* method), 323
- `__bool__` () (*rti.connexdds.Int16Seq* method), 314
- `__bool__` () (*rti.connexdds.Int32Seq* method), 317
- `__bool__` () (*rti.connexdds.Int64Seq* method), 320
- `__bool__` () (*rti.connexdds.LocatorFilterElementSeq* method), 333
- `__bool__` () (*rti.connexdds.LocatorSeq* method), 338
- `__bool__` () (*rti.connexdds.MemberSeq* method), 347
- `__bool__` () (*rti.connexdds.MonitoringMetricSelectionSeq* method), 353
- `__bool__` () (*rti.connexdds.MulticastMappingSeq* method), 358
- `__bool__` () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 371
- `__bool__` () (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 381
- `__bool__` () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 392
- `__bool__` () (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 406
- `__bool__` () (*rti.connexdds.ParticipantBuiltinTopicData-TimestampedSeq* method), 409
- `__bool__` () (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 401
- `__bool__` () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 427
- `__bool__` () (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 437
- `__bool__` () (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 449
- `__bool__` () (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 464
- `__bool__` () (*rti.connexdds.PublicationBuiltinTopicData-TimestampedSeq* method), 466
- `__bool__` () (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 458
- `__bool__` () (*rti.connexdds.PublisherSeq* method), 476
- `__bool__` () (*rti.connexdds.QosPolicyCountSeq* method), 478
- `__bool__` () (*rti.connexdds.RtpsReservedPortKindMask* method), 501
- `__bool__` () (*rti.connexdds.SampleFlag* method), 505
- `__bool__` () (*rti.connexdds.SampleLostState* method), 511
- `__bool__` () (*rti.connexdds.SampleRejectedState* method), 514
- `__bool__` () (*rti.connexdds.SampleState* method), 516
- `__bool__` () (*rti.connexdds.ServiceRequest.ContentFiltered-TopicSeq* method), 525
- `__bool__` () (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 534
- `__bool__` () (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 545
- `__bool__` () (*rti.connexdds.ServiceRequestSeq* method), 560
- `__bool__` () (*rti.connexdds.ServiceRequestTimestampedSeq* method), 562
- `__bool__` () (*rti.connexdds.ServiceRequest.TopicSeq* method), 554
- `__bool__` () (*rti.connexdds.StatusMask* method), 567
- `__bool__` () (*rti.connexdds.StreamKind* method), 569
- `__bool__` () (*rti.connexdds.StringMap* method), 572
- `__bool__` () (*rti.connexdds.StringPairSeq* method), 572
- `__bool__` () (*rti.connexdds.StringSeq* method), 575
- `__bool__` () (*rti.connexdds.SubscriberSeq* method), 582
- `__bool__` () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 586
- `__bool__` () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 596
- `__bool__` () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 607
- `__bool__` () (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 622
- `__bool__` () (*rti.connexdds.SubscriptionBuiltinTopicData-TimestampedSeq* method), 625
- `__bool__` () (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 616
- `__bool__` () (*rti.connexdds.ThreadSettingsKindMask* method), 633
- `__bool__` () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 641
- `__bool__` () (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq* method), 651
- `__bool__` () (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq* method), 662
- `__bool__` () (*rti.connexdds.TopicBuiltinTopicDataSeq* method), 676
- `__bool__` () (*rti.connexdds.TopicBuiltinTopicData-TimestampedSeq* method), 678
- `__bool__` () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq* method), 671
- `__bool__` () (*rti.connexdds.TopicSeq* method), 690
- `__bool__` () (*rti.connexdds.TransportBuiltinMask* method), 693
- `__bool__` () (*rti.connexdds.TransportInfoSeq* method), 699
- `__bool__` () (*rti.connexdds.TransportMulticastSeq* method),

- 706
- `__bool__()` (*rti.connexdds.TransportMulticastSettingsSeq method*), 709
 - `__bool__()` (*rti.connexdds.TransportUnicastSettingsSeq method*), 713
 - `__bool__()` (*rti.connexdds.Uint8Seq method*), 734
 - `__bool__()` (*rti.connexdds.Uint16Seq method*), 726
 - `__bool__()` (*rti.connexdds.Uint32Seq method*), 729
 - `__bool__()` (*rti.connexdds.Uint64Seq method*), 731
 - `__bool__()` (*rti.connexdds.UnionMemberSeq method*), 738
 - `__bool__()` (*rti.connexdds.ViewState method*), 745
 - `__bool__()` (*rti.connexdds.WcharSeq method*), 749
 - `__bool__()` (*rti.connexdds.WstringSeq method*), 757
 - `__contains__()` (*rti.connexdds.ActivityContextMask method*), 38
 - `__contains__()` (*rti.connexdds.AnyDataReaderSeq method*), 43
 - `__contains__()` (*rti.connexdds.AnyDataWriterSeq method*), 46
 - `__contains__()` (*rti.connexdds.AnyTopicSeq method*), 49
 - `__contains__()` (*rti.connexdds.BoolSeq method*), 54
 - `__contains__()` (*rti.connexdds.ChannelSettingsSeq method*), 63
 - `__contains__()` (*rti.connexdds.CharSeq method*), 66
 - `__contains__()` (*rti.connexdds.CompressionIdMask method*), 70
 - `__contains__()` (*rti.connexdds.ConditionSeq method*), 74
 - `__contains__()` (*rti.connexdds.ContentFilteredTopicSeq method*), 77
 - `__contains__()` (*rti.connexdds.CookieSeq method*), 80
 - `__contains__()` (*rti.connexdds.DataReaderSeq method*), 107
 - `__contains__()` (*rti.connexdds.DataTag method*), 113
 - `__contains__()` (*rti.connexdds.DataWriterSeq method*), 138
 - `__contains__()` (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method*), 151
 - `__contains__()` (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method*), 154
 - `__contains__()` (*rti.connexdds.DomainParticipantSeq method*), 176
 - `__contains__()` (*rti.connexdds.DynamicData method*), 220
 - `__contains__()` (*rti.connexdds.DynamicData.ContentFilteredTopicSeq method*), 186
 - `__contains__()` (*rti.connexdds.DynamicData.DataReaderSeq method*), 195
 - `__contains__()` (*rti.connexdds.DynamicData.DataWriterSeq method*), 206
 - `__contains__()` (*rti.connexdds.DynamicData.FieldsView method*), 208
 - `__contains__()` (*rti.connexdds.DynamicData.ItemsView method*), 210
 - `__contains__()` (*rti.connexdds.DynamicDataSeq method*), 249
 - `__contains__()` (*rti.connexdds.DynamicData.TimestampedSeq method*), 252
 - `__contains__()` (*rti.connexdds.DynamicData.TopicSeq method*), 216
 - `__contains__()` (*rti.connexdds.EndpointGroupSeq method*), 256
 - `__contains__()` (*rti.connexdds.EntitySeq method*), 261
 - `__contains__()` (*rti.connexdds.EnumMemberSeq method*), 264
 - `__contains__()` (*rti.connexdds.Float32Seq method*), 274
 - `__contains__()` (*rti.connexdds.Float64Seq method*), 277
 - `__contains__()` (*rti.connexdds.Float128Seq method*), 272
 - `__contains__()` (*rti.connexdds.IAnyDataReaderSeq method*), 291
 - `__contains__()` (*rti.connexdds.IAnyDataWriterSeq method*), 294
 - `__contains__()` (*rti.connexdds.IAnyTopicSeq method*), 297
 - `__contains__()` (*rti.connexdds.IConditionSeq method*), 299
 - `__contains__()` (*rti.connexdds.IEntitySeq method*), 303
 - `__contains__()` (*rti.connexdds.InstanceHandleSeq method*), 309
 - `__contains__()` (*rti.connexdds.InstanceState method*), 311
 - `__contains__()` (*rti.connexdds.Int8Seq method*), 323
 - `__contains__()` (*rti.connexdds.Int16Seq method*), 314
 - `__contains__()` (*rti.connexdds.Int32Seq method*), 317
 - `__contains__()` (*rti.connexdds.Int64Seq method*), 321
 - `__contains__()` (*rti.connexdds.LocatorFilterElementSeq method*), 333
 - `__contains__()` (*rti.connexdds.LocatorSeq method*), 338
 - `__contains__()` (*rti.connexdds.MemberSeq method*), 347
 - `__contains__()` (*rti.connexdds.MonitoringMetricSelectionSeq method*), 353
 - `__contains__()` (*rti.connexdds.MulticastMappingSeq method*), 358
 - `__contains__()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 371
 - `__contains__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 381
 - `__contains__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 392
 - `__contains__()` (*rti.connexdds.ParticipantBuiltinTopicDataSeq method*), 406
 - `__contains__()` (*rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq method*), 409
 - `__contains__()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq method*), 401
 - `__contains__()` (*rti.connexdds.Property method*), 424
 - `__contains__()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 427

- `__contains__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 437
- `__contains__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 449
- `__contains__()` (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 464
- `__contains__()` (*rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq* method), 466
- `__contains__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 458
- `__contains__()` (*rti.connexdds.PublisherSeq* method), 476
- `__contains__()` (*rti.connexdds.QosPolicyCountSeq* method), 478
- `__contains__()` (*rti.connexdds.RtpsReservedPortKindMask* method), 501
- `__contains__()` (*rti.connexdds.SampleFlag* method), 505
- `__contains__()` (*rti.connexdds.SampleLostState* method), 511
- `__contains__()` (*rti.connexdds.SampleRejectedState* method), 514
- `__contains__()` (*rti.connexdds.SampleState* method), 517
- `__contains__()` (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq* method), 525
- `__contains__()` (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 534
- `__contains__()` (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 545
- `__contains__()` (*rti.connexdds.ServiceRequestSeq* method), 560
- `__contains__()` (*rti.connexdds.ServiceRequestTimestampedSeq* method), 563
- `__contains__()` (*rti.connexdds.ServiceRequest.TopicSeq* method), 554
- `__contains__()` (*rti.connexdds.StatusMask* method), 567
- `__contains__()` (*rti.connexdds.StreamKind* method), 569
- `__contains__()` (*rti.connexdds.StringMap* method), 572
- `__contains__()` (*rti.connexdds.StringPairSeq* method), 573
- `__contains__()` (*rti.connexdds.StringSeq* method), 575
- `__contains__()` (*rti.connexdds.SubscriberSeq* method), 582
- `__contains__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 586
- `__contains__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 596
- `__contains__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 607
- `__contains__()` (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 622
- `__contains__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TimestampedSeq* method), 625
- `__contains__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 616
- `__contains__()` (*rti.connexdds.ThreadSettingsKindMask* method), 633
- `__contains__()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 641
- `__contains__()` (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq* method), 651
- `__contains__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq* method), 662
- `__contains__()` (*rti.connexdds.TopicBuiltinTopicDataSeq* method), 676
- `__contains__()` (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq* method), 678
- `__contains__()` (*rti.connexdds.TopicBuiltinTopicData.TopicSeq* method), 671
- `__contains__()` (*rti.connexdds.TopicSeq* method), 690
- `__contains__()` (*rti.connexdds.TransportBuiltinMask* method), 693
- `__contains__()` (*rti.connexdds.TransportInfoSeq* method), 699
- `__contains__()` (*rti.connexdds.TransportMulticastSeq* method), 706
- `__contains__()` (*rti.connexdds.TransportMulticastSettingsSeq* method), 709
- `__contains__()` (*rti.connexdds.TransportUnicastSettingsSeq* method), 713
- `__contains__()` (*rti.connexdds.TriggeredConditions* method), 716
- `__contains__()` (*rti.connexdds.Uint8Seq* method), 734
- `__contains__()` (*rti.connexdds.Uint16Seq* method), 726
- `__contains__()` (*rti.connexdds.Uint32Seq* method), 729
- `__contains__()` (*rti.connexdds.Uint64Seq* method), 731
- `__contains__()` (*rti.connexdds.UnionMemberSeq* method), 738
- `__contains__()` (*rti.connexdds.ViewState* method), 745
- `__contains__()` (*rti.connexdds.WcharSeq* method), 749
- `__contains__()` (*rti.connexdds.WstringSeq* method), 757
- `__delitem__()` (*rti.connexdds.AnyDataReaderSeq* method), 43
- `__delitem__()` (*rti.connexdds.AnyDataWriterSeq* method), 46
- `__delitem__()` (*rti.connexdds.AnyTopicSeq* method), 49
- `__delitem__()` (*rti.connexdds.BoolSeq* method), 54
- `__delitem__()` (*rti.connexdds.ChannelSettingsSeq* method), 63
- `__delitem__()` (*rti.connexdds.CharSeq* method), 66
- `__delitem__()` (*rti.connexdds.ConditionSeq* method), 74

- `__delitem__()` (*rti.connexdds.ContentFilteredTopicSeq method*), 77
- `__delitem__()` (*rti.connexdds.CookieSeq method*), 80
- `__delitem__()` (*rti.connexdds.DataReaderSeq method*), 107
- `__delitem__()` (*rti.connexdds.DataWriterSeq method*), 138
- `__delitem__()` (*rti.connexdds.DomainParticipantSeq method*), 176
- `__delitem__()` (*rti.connexdds.DynamicData.ContentFilteredTopicSeq method*), 186
- `__delitem__()` (*rti.connexdds.DynamicData.DataReaderSeq method*), 195
- `__delitem__()` (*rti.connexdds.DynamicData.DataWriterSeq method*), 206
- `__delitem__()` (*rti.connexdds.DynamicDataSeq method*), 249
- `__delitem__()` (*rti.connexdds.DynamicDataTimestampedSeq method*), 252
- `__delitem__()` (*rti.connexdds.DynamicData.TopicSeq method*), 216
- `__delitem__()` (*rti.connexdds.EndpointGroupSeq method*), 256
- `__delitem__()` (*rti.connexdds.EntitySeq method*), 261
- `__delitem__()` (*rti.connexdds.EnumMemberSeq method*), 264
- `__delitem__()` (*rti.connexdds.Float32Seq method*), 274
- `__delitem__()` (*rti.connexdds.Float64Seq method*), 277
- `__delitem__()` (*rti.connexdds.Float128Seq method*), 272
- `__delitem__()` (*rti.connexdds.IAnyDataReaderSeq method*), 291
- `__delitem__()` (*rti.connexdds.IAnyDataWriterSeq method*), 294
- `__delitem__()` (*rti.connexdds.IAnyTopicSeq method*), 297
- `__delitem__()` (*rti.connexdds.IConditionSeq method*), 299
- `__delitem__()` (*rti.connexdds.IEntitySeq method*), 303
- `__delitem__()` (*rti.connexdds.InstanceHandleSeq method*), 309
- `__delitem__()` (*rti.connexdds.Int8Seq method*), 323
- `__delitem__()` (*rti.connexdds.Int16Seq method*), 314
- `__delitem__()` (*rti.connexdds.Int32Seq method*), 317
- `__delitem__()` (*rti.connexdds.Int64Seq method*), 321
- `__delitem__()` (*rti.connexdds.LocatorFilterElementSeq method*), 333
- `__delitem__()` (*rti.connexdds.LocatorSeq method*), 338
- `__delitem__()` (*rti.connexdds.MemberSeq method*), 347
- `__delitem__()` (*rti.connexdds.MonitoringMetricSelectionSeq method*), 353
- `__delitem__()` (*rti.connexdds.MulticastMappingSeq method*), 358
- `__delitem__()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 371
- `__delitem__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 381
- `__delitem__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 393
- `__delitem__()` (*rti.connexdds.ParticipantBuiltinTopicDataSeq method*), 406
- `__delitem__()` (*rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 409
- `__delitem__()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq method*), 401
- `__delitem__()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 427
- `__delitem__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq method*), 437
- `__delitem__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq method*), 449
- `__delitem__()` (*rti.connexdds.PublicationBuiltinTopicDataSeq method*), 464
- `__delitem__()` (*rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq method*), 466
- `__delitem__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq method*), 458
- `__delitem__()` (*rti.connexdds.PublisherSeq method*), 476
- `__delitem__()` (*rti.connexdds.QosPolicyCountSeq method*), 478
- `__delitem__()` (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq method*), 525
- `__delitem__()` (*rti.connexdds.ServiceRequest.DataReaderSeq method*), 534
- `__delitem__()` (*rti.connexdds.ServiceRequest.DataWriterSeq method*), 545
- `__delitem__()` (*rti.connexdds.ServiceRequestSeq method*), 560
- `__delitem__()` (*rti.connexdds.ServiceRequestTimestampedSeq method*), 563
- `__delitem__()` (*rti.connexdds.ServiceRequest.TopicSeq method*), 554
- `__delitem__()` (*rti.connexdds.StringMap method*), 572
- `__delitem__()` (*rti.connexdds.StringPairSeq method*), 573
- `__delitem__()` (*rti.connexdds.StringSeq method*), 575
- `__delitem__()` (*rti.connexdds.SubscriberSeq method*), 582
- `__delitem__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 586
- `__delitem__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 596
- `__delitem__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 596

607
 __delitem__() (rti.connexdds.SubscriptionBuiltinTopicDataSeq method), 622
 __delitem__() (rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq method), 625
 __delitem__() (rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq method), 616
 __delitem__() (rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method), 641
 __delitem__() (rti.connexdds.TopicBuiltinTopicData.DataReaderSeq method), 651
 __delitem__() (rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method), 662
 __delitem__() (rti.connexdds.TopicBuiltinTopicDataSeq method), 676
 __delitem__() (rti.connexdds.TopicBuiltinTopicDataTimestampedSeq method), 678
 __delitem__() (rti.connexdds.TopicBuiltinTopicData.TopicSeq method), 671
 __delitem__() (rti.connexdds.TopicSeq method), 690
 __delitem__() (rti.connexdds.TransportInfoSeq method), 699
 __delitem__() (rti.connexdds.TransportMulticastSeq method), 706
 __delitem__() (rti.connexdds.TransportMulticastSettingsSeq method), 709
 __delitem__() (rti.connexdds.TransportUnicastSettingsSeq method), 713
 __delitem__() (rti.connexdds.Uint8Seq method), 734
 __delitem__() (rti.connexdds.Uint16Seq method), 726
 __delitem__() (rti.connexdds.Uint32Seq method), 729
 __delitem__() (rti.connexdds.Uint64Seq method), 732
 __delitem__() (rti.connexdds.UnionMemberSeq method), 738
 __delitem__() (rti.connexdds.WcharSeq method), 749
 __delitem__() (rti.connexdds.WstringSeq method), 757
 __enter__() (rti.connexdds.CoherentAccess method), 68
 __enter__() (rti.connexdds.CoherentSet method), 68
 __enter__() (rti.connexdds.DataReader.LoanedSamples method), 83
 __enter__() (rti.connexdds.DataWriter method), 114
 __enter__() (rti.connexdds.DomainParticipant method), 157
 __enter__() (rti.connexdds.DynamicData.DataReader method), 189
 __enter__() (rti.connexdds.DynamicData.DataWriter method), 197
 __enter__() (rti.connexdds.DynamicData.LoanedSamples method), 210
 __enter__() (rti.connexdds.DynamicData.ValidLoanedSamples method), 219
 __enter__() (rti.connexdds.IReadCondition method), 305
 __enter__() (rti.connexdds.LoanedDynamicData method), 331
 __enter__() (rti.connexdds.ParticipantBuiltinTopicData.DataReader method), 375
 __enter__() (rti.connexdds.ParticipantBuiltinTopicData.DataWriter method), 383
 __enter__() (rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples method), 395
 __enter__() (rti.connexdds.ParticipantBuiltinTopicData.ValidLoanedSamples method), 403
 __enter__() (rti.connexdds.PublicationBuiltinTopicData.DataReader method), 431
 __enter__() (rti.connexdds.PublicationBuiltinTopicData.DataWriter method), 439
 __enter__() (rti.connexdds.PublicationBuiltinTopicData.LoanedSamples method), 452
 __enter__() (rti.connexdds.PublicationBuiltinTopicData.ValidLoanedSamples method), 460
 __enter__() (rti.connexdds.ServiceRequest.DataReader method), 528
 __enter__() (rti.connexdds.ServiceRequest.DataWriter method), 536
 __enter__() (rti.connexdds.ServiceRequest.LoanedSamples method), 548
 __enter__() (rti.connexdds.ServiceRequest.ValidLoanedSamples method), 556
 __enter__() (rti.connexdds.SubscriptionBuiltinTopicData.DataReader method), 589
 __enter__() (rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method), 598
 __enter__() (rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples method), 610
 __enter__() (rti.connexdds.SubscriptionBuiltinTopicData.ValidLoanedSamples method), 618
 __enter__() (rti.connexdds.SuspendedPublication method), 628
 __enter__() (rti.connexdds.ThreadContext method), 630
 __enter__() (rti.connexdds.TopicBuiltinTopicData.DataReader method), 645
 __enter__() (rti.connexdds.TopicBuiltinTopicData.DataWriter method), 653

- `__enter__()` (*rti.connexdds.TopicBuiltinTopicData.LoanedSamples method*), 665
- `__enter__()` (*rti.connexdds.TopicBuiltinTopicData.ValidLoanedSamples method*), 673
- `__enter__()` (*rti.connexdds.TopicQuery method*), 686
- `__eq__()` (*rti.connexdds.AcknowledgmentKind method*), 36
- `__eq__()` (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind method*), 36
- `__eq__()` (*rti.connexdds.ACTEnumMember method*), 31
- `__eq__()` (*rti.connexdds.ActivityContextMask method*), 38
- `__eq__()` (*rti.connexdds.ACTMember method*), 32
- `__eq__()` (*rti.connexdds.ACTUnionMember method*), 33
- `__eq__()` (*rti.connexdds.AliasType method*), 40
- `__eq__()` (*rti.connexdds.AllocationSettings method*), 41
- `__eq__()` (*rti.connexdds.AnyDataReaderSeq method*), 43
- `__eq__()` (*rti.connexdds.AnyDataWriterSeq method*), 46
- `__eq__()` (*rti.connexdds.AnyTopicSeq method*), 49
- `__eq__()` (*rti.connexdds.ArrayType method*), 51
- `__eq__()` (*rti.connexdds.AsynchronousPublisher method*), 51
- `__eq__()` (*rti.connexdds.Availability method*), 52
- `__eq__()` (*rti.connexdds.Batch method*), 53
- `__eq__()` (*rti.connexdds.BoolSeq method*), 54
- `__eq__()` (*rti.connexdds.BoolType method*), 56
- `__eq__()` (*rti.connexdds.BuiltinTopicKey method*), 58
- `__eq__()` (*rti.connexdds.BuiltinTopicReaderResourceLimits method*), 59
- `__eq__()` (*rti.connexdds.ByteVector method*), 60
- `__eq__()` (*rti.connexdds.CdrPaddingKind method*), 62
- `__eq__()` (*rti.connexdds.CdrPaddingKind.CdrPaddingKind method*), 61
- `__eq__()` (*rti.connexdds.ChannelSettings method*), 63
- `__eq__()` (*rti.connexdds.ChannelSettingsSeq method*), 64
- `__eq__()` (*rti.connexdds.CharSeq method*), 66
- `__eq__()` (*rti.connexdds.CoherentSetInfo method*), 68
- `__eq__()` (*rti.connexdds.CollectionType method*), 69
- `__eq__()` (*rti.connexdds.CompressionIdMask method*), 70
- `__eq__()` (*rti.connexdds.CompressionSettings method*), 73
- `__eq__()` (*rti.connexdds.ConditionSeq method*), 74
- `__eq__()` (*rti.connexdds.ContentFilteredTopic method*), 76
- `__eq__()` (*rti.connexdds.ContentFilteredTopicSeq method*), 77
- `__eq__()` (*rti.connexdds.ContentFilterProperty method*), 76
- `__eq__()` (*rti.connexdds.Cookie method*), 79
- `__eq__()` (*rti.connexdds.CookieSeq method*), 80
- `__eq__()` (*rti.connexdds.CookieVector method*), 82
- `__eq__()` (*rti.connexdds.Database method*), 140
- `__eq__()` (*rti.connexdds.DataReader method*), 85
- `__eq__()` (*rti.connexdds.DataReaderInstanceRemovalKind method*), 93
- `__eq__()` (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 92
- `__eq__()` (*rti.connexdds.DataReaderProtocol method*), 95
- `__eq__()` (*rti.connexdds.DataReaderQos method*), 97
- `__eq__()` (*rti.connexdds.DataReaderResourceLimits method*), 104
- `__eq__()` (*rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings method*), 106
- `__eq__()` (*rti.connexdds.DataReaderSeq method*), 107
- `__eq__()` (*rti.connexdds.DataRepresentation method*), 109
- `__eq__()` (*rti.connexdds.DataState method*), 110
- `__eq__()` (*rti.connexdds.DataStateEx method*), 111
- `__eq__()` (*rti.connexdds.DataTag method*), 113
- `__eq__()` (*rti.connexdds.DataWriter method*), 114
- `__eq__()` (*rti.connexdds.DataWriterProtocol method*), 122
- `__eq__()` (*rti.connexdds.DataWriterQos method*), 124
- `__eq__()` (*rti.connexdds.DataWriterResourceLimits method*), 132
- `__eq__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind method*), 136
- `__eq__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 135
- `__eq__()` (*rti.connexdds.DataWriterSeq method*), 138
- `__eq__()` (*rti.connexdds.DataWriterShmemRefTransferModeSettings method*), 140
- `__eq__()` (*rti.connexdds.DataWriterTransferMode method*), 140
- `__eq__()` (*rti.connexdds.Deadline method*), 141
- `__eq__()` (*rti.connexdds.DestinationOrder method*), 142
- `__eq__()` (*rti.connexdds.DestinationOrderKind method*), 143
- `__eq__()` (*rti.connexdds.DestinationOrderKind.DestinationOrderKind method*), 143
- `__eq__()` (*rti.connexdds.DestinationOrderScopeKind method*), 146
- `__eq__()` (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 145
- `__eq__()` (*rti.connexdds.Discovery method*), 147
- `__eq__()` (*rti.connexdds.DiscoveryConfig method*), 147
- `__eq__()` (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method*), 151
- `__eq__()` (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method*), 154
- `__eq__()` (*rti.connexdds.DomainParticipant method*), 157
- `__eq__()` (*rti.connexdds.DomainParticipantConfigParams method*), 163
- `__eq__()` (*rti.connexdds.DomainParticipantFactoryQos method*), 164
- `__eq__()` (*rti.connexdds.DomainParticipantQos method*), 166
- `__eq__()` (*rti.connexdds.DomainParticipantResourceLimits method*), 171
- `__eq__()` (*rti.connexdds.DomainParticipantSeq method*), 176
- `__eq__()` (*rti.connexdds.Durability method*), 178
- `__eq__()` (*rti.connexdds.DurabilityKind method*), 180
- `__eq__()` (*rti.connexdds.DurabilityKind.DurabilityKind method*), 180
- `__eq__()` (*rti.connexdds.DurabilityService method*), 181

- `__eq__` () (*rti.connexdds.Duration* method), 182
- `__eq__` () (*rti.connexdds.DynamicData* method), 220
- `__eq__` () (*rti.connexdds.DynamicData.ContentFilteredTopic* method), 185
- `__eq__` () (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 186
- `__eq__` () (*rti.connexdds.DynamicData.DataReader* method), 189
- `__eq__` () (*rti.connexdds.DynamicData.DataReaderSeq* method), 195
- `__eq__` () (*rti.connexdds.DynamicData.DataWriter* method), 197
- `__eq__` () (*rti.connexdds.DynamicData.DataWriterSeq* method), 206
- `__eq__` () (*rti.connexdds.DynamicDataEncapsulationKind* method), 246
- `__eq__` () (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind* method), 245
- `__eq__` () (*rti.connexdds.DynamicDataInfo* method), 247
- `__eq__` () (*rti.connexdds.DynamicDataMemberInfo* method), 248
- `__eq__` () (*rti.connexdds.DynamicDataProperty* method), 249
- `__eq__` () (*rti.connexdds.DynamicDataSeq* method), 250
- `__eq__` () (*rti.connexdds.DynamicDataTimestampedSeq* method), 252
- `__eq__` () (*rti.connexdds.DynamicData.Topic* method), 214
- `__eq__` () (*rti.connexdds.DynamicData.TopicDescription* method), 215
- `__eq__` () (*rti.connexdds.DynamicData.TopicSeq* method), 217
- `__eq__` () (*rti.connexdds.DynamicDataTypeSerializationProperty* method), 254
- `__eq__` () (*rti.connexdds.DynamicType* method), 255
- `__eq__` () (*rti.connexdds.DynamicTypePrintFormatProperty* method), 256
- `__eq__` () (*rti.connexdds.EndpointGroupSeq* method), 257
- `__eq__` () (*rti.connexdds.EndpointGroupVector* method), 258
- `__eq__` () (*rti.connexdds.EntityFactory* method), 260
- `__eq__` () (*rti.connexdds.EntityName* method), 260
- `__eq__` () (*rti.connexdds.EntitySeq* method), 261
- `__eq__` () (*rti.connexdds.EnumMember* method), 263
- `__eq__` () (*rti.connexdds.EnumMemberSeq* method), 264
- `__eq__` () (*rti.connexdds.EnumType* method), 266
- `__eq__` () (*rti.connexdds.Event* method), 266
- `__eq__` () (*rti.connexdds.ExclusiveArea* method), 268
- `__eq__` () (*rti.connexdds.ExpressionProperty* method), 268
- `__eq__` () (*rti.connexdds.ExtensibilityKind* method), 270
- `__eq__` () (*rti.connexdds.ExtensibilityKind.ExtensibilityKind* method), 269
- `__eq__` () (*rti.connexdds.FilterSampleInfo* method), 271
- `__eq__` () (*rti.connexdds.Float32Seq* method), 275
- `__eq__` () (*rti.connexdds.Float32Type* method), 276
- `__eq__` () (*rti.connexdds.Float64Seq* method), 277
- `__eq__` () (*rti.connexdds.Float64Type* method), 279
- `__eq__` () (*rti.connexdds.Float128Seq* method), 272
- `__eq__` () (*rti.connexdds.Float128Type* method), 274
- `__eq__` () (*rti.connexdds.FlowController* method), 280
- `__eq__` () (*rti.connexdds.FlowControllerProperty* method), 280
- `__eq__` () (*rti.connexdds.FlowControllerSchedulingPolicy* method), 283
- `__eq__` () (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy* method), 282
- `__eq__` () (*rti.connexdds.FlowControllerTokenBucketProperty* method), 284
- `__eq__` () (*rti.connexdds.GroupData* method), 285
- `__eq__` () (*rti.connexdds.GuardCondition* method), 286
- `__eq__` () (*rti.connexdds.Guid* method), 287
- `__eq__` () (*rti.connexdds.History* method), 288
- `__eq__` () (*rti.connexdds.HistoryKind* method), 289
- `__eq__` () (*rti.connexdds.HistoryKind.HistoryKind* method), 289
- `__eq__` () (*rti.connexdds.IAnyDataReader* method), 290
- `__eq__` () (*rti.connexdds.IAnyDataReaderSeq* method), 291
- `__eq__` () (*rti.connexdds.IAnyDataWriter* method), 293
- `__eq__` () (*rti.connexdds.IAnyDataWriterSeq* method), 294
- `__eq__` () (*rti.connexdds.IAnyTopic* method), 296
- `__eq__` () (*rti.connexdds.IAnyTopicSeq* method), 297
- `__eq__` () (*rti.connexdds.ICondition* method), 299
- `__eq__` () (*rti.connexdds.IConditionSeq* method), 300
- `__eq__` () (*rti.connexdds.IEntity* method), 302
- `__eq__` () (*rti.connexdds.IEntitySeq* method), 303
- `__eq__` () (*rti.connexdds.IgnoredEntityReplacementKind* method), 307
- `__eq__` () (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind* method), 306
- `__eq__` () (*rti.connexdds.InstanceHandle* method), 308
- `__eq__` () (*rti.connexdds.InstanceHandleSeq* method), 309
- `__eq__` () (*rti.connexdds.InstanceState* method), 311
- `__eq__` () (*rti.connexdds.InstanceStateConsistencyKind* method), 314
- `__eq__` () (*rti.connexdds.Int8Seq* method), 323
- `__eq__` () (*rti.connexdds.Int8Type* method), 325
- `__eq__` () (*rti.connexdds.Int16Seq* method), 315
- `__eq__` () (*rti.connexdds.Int16Type* method), 316
- `__eq__` () (*rti.connexdds.Int32Seq* method), 317
- `__eq__` () (*rti.connexdds.Int32Type* method), 319
- `__eq__` () (*rti.connexdds.Int32Vector* method), 319
- `__eq__` () (*rti.connexdds.Int64Seq* method), 321
- `__eq__` () (*rti.connexdds.Int64Type* method), 323
- `__eq__` () (*rti.connexdds.IReadCondition* method), 305
- `__eq__` () (*rti.connexdds.LatencyBudget* method), 326
- `__eq__` () (*rti.connexdds.Lifespan* method), 326
- `__eq__` () (*rti.connexdds.Liveliness* method), 327
- `__eq__` () (*rti.connexdds.LivelinessKind* method), 330
- `__eq__` () (*rti.connexdds.LivelinessKind.LivelinessKind* method), 329
- `__eq__` () (*rti.connexdds.Locator* method), 331
- `__eq__` () (*rti.connexdds.LocatorFilter* method), 332
- `__eq__` () (*rti.connexdds.LocatorFilterElement* method), 332
- `__eq__` () (*rti.connexdds.LocatorFilterElementSeq* method), 333
- `__eq__` () (*rti.connexdds.LocatorKind* method), 337

- `__eq__()` (*rti.connexdds.LocatorKind.LocatorKind method*), 336
- `__eq__()` (*rti.connexdds.LocatorSeq method*), 339
- `__eq__()` (*rti.connexdds.LocatorVector method*), 340
- `__eq__()` (*rti.connexdds.LogCategory method*), 343
- `__eq__()` (*rti.connexdds.LogCategory.LogCategory method*), 342
- `__eq__()` (*rti.connexdds.LongDouble method*), 345
- `__eq__()` (*rti.connexdds.Member method*), 346
- `__eq__()` (*rti.connexdds.MemberSeq method*), 347
- `__eq__()` (*rti.connexdds.Monitoring method*), 349
- `__eq__()` (*rti.connexdds.MonitoringDedicatedParticipantSettings method*), 349
- `__eq__()` (*rti.connexdds.MonitoringDistributionSettings method*), 350
- `__eq__()` (*rti.connexdds.MonitoringEventDistributionSettings method*), 351
- `__eq__()` (*rti.connexdds.MonitoringLoggingDistributionSettings method*), 351
- `__eq__()` (*rti.connexdds.MonitoringLoggingForwardingSettings method*), 352
- `__eq__()` (*rti.connexdds.MonitoringMetricSelection method*), 353
- `__eq__()` (*rti.connexdds.MonitoringMetricSelectionSeq method*), 354
- `__eq__()` (*rti.connexdds.MonitoringPeriodicDistributionSettings method*), 356
- `__eq__()` (*rti.connexdds.MonitoringTelemetryData method*), 356
- `__eq__()` (*rti.connexdds.MulticastMapping method*), 357
- `__eq__()` (*rti.connexdds.MulticastMappingSeq method*), 358
- `__eq__()` (*rti.connexdds.MultiChannel method*), 357
- `__eq__()` (*rti.connexdds.Ownership method*), 367
- `__eq__()` (*rti.connexdds.OwnershipKind method*), 368
- `__eq__()` (*rti.connexdds.OwnershipKind.OwnershipKind method*), 368
- `__eq__()` (*rti.connexdds.OwnershipStrength method*), 369
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData method*), 405
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic method*), 370
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 371
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*), 375
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 381
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*), 383
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 393
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicDataSeq method*), 407
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq method*), 409
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData.Topic method*), 400
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicDescription method*), 400
- `__eq__()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq method*), 402
- `__eq__()` (*rti.connexdds.Partition method*), 411
- `__eq__()` (*rti.connexdds.PersistentJournalKind method*), 412
- `__eq__()` (*rti.connexdds.PersistentStorageSettings method*), 413
- `__eq__()` (*rti.connexdds.PersistentSynchronizationKind method*), 414
- `__eq__()` (*rti.connexdds.Presentation method*), 415
- `__eq__()` (*rti.connexdds.PresentationAccessScopeKind method*), 417
- `__eq__()` (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 416
- `__eq__()` (*rti.connexdds.PrintFormat method*), 420
- `__eq__()` (*rti.connexdds.PrintFormatKind method*), 421
- `__eq__()` (*rti.connexdds.PrintFormatKind.PrintFormatKind method*), 421
- `__eq__()` (*rti.connexdds.PrintFormat.PrintFormat method*), 419
- `__eq__()` (*rti.connexdds.ProductVersion method*), 423
- `__eq__()` (*rti.connexdds.Property method*), 424
- `__eq__()` (*rti.connexdds.ProtocolVersion method*), 425
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData method*), 461
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic method*), 426
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 428
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 431
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq method*), 437
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*), 439
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq method*), 449
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicDataSeq method*), 464
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData.TimestampedSeq method*),

- 467
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData.Topic* method), 456
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicDescription* method), 457
- `__eq__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 458
- `__eq__()` (*rti.connexdds.Publisher* method), 472
- `__eq__()` (*rti.connexdds.PublisherQos* method), 474
- `__eq__()` (*rti.connexdds.PublisherSeq* method), 476
- `__eq__()` (*rti.connexdds.PublishMode* method), 469
- `__eq__()` (*rti.connexdds.PublishModeKind* method), 471
- `__eq__()` (*rti.connexdds.PublishModeKind.PublishModeKind* method), 471
- `__eq__()` (*rti.connexdds.QosPolicyCount* method), 478
- `__eq__()` (*rti.connexdds.QosPolicyCountSeq* method), 479
- `__eq__()` (*rti.connexdds.QosPrintFormat* method), 480
- `__eq__()` (*rti.connexdds.QosProvider* method), 481
- `__eq__()` (*rti.connexdds.QosProviderParams* method), 484
- `__eq__()` (*rti.connexdds.Query* method), 485
- `__eq__()` (*rti.connexdds.QueryCondition* method), 486
- `__eq__()` (*rti.connexdds.ReaderDataLifecycle* method), 487
- `__eq__()` (*rti.connexdds.ReceiverPool* method), 489
- `__eq__()` (*rti.connexdds.Reliability* method), 489
- `__eq__()` (*rti.connexdds.ReliabilityKind* method), 491
- `__eq__()` (*rti.connexdds.ReliabilityKind.ReliabilityKind* method), 491
- `__eq__()` (*rti.connexdds.RemoteParticipantPurgeKind* method), 494
- `__eq__()` (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* method), 494
- `__eq__()` (*rti.connexdds.ResourceLimits* method), 496
- `__eq__()` (*rti.connexdds.RtpsReliableReaderProtocol* method), 497
- `__eq__()` (*rti.connexdds.RtpsReliableWriterProtocol* method), 498
- `__eq__()` (*rti.connexdds.RtpsReservedPortKindMask* method), 501
- `__eq__()` (*rti.connexdds.RtpsWellKnownPorts* method), 504
- `__eq__()` (*rti.connexdds.SampleFlag* method), 505
- `__eq__()` (*rti.connexdds.SampleIdentity* method), 507
- `__eq__()` (*rti.connexdds.SampleInfo* method), 508
- `__eq__()` (*rti.connexdds.SampleLostState* method), 511
- `__eq__()` (*rti.connexdds.SampleRejectedState* method), 514
- `__eq__()` (*rti.connexdds.SampleState* method), 517
- `__eq__()` (*rti.connexdds.SequenceNumber* method), 519
- `__eq__()` (*rti.connexdds.SequenceType* method), 520
- `__eq__()` (*rti.connexdds.Service* method), 521
- `__eq__()` (*rti.connexdds.ServiceKind* method), 523
- `__eq__()` (*rti.connexdds.ServiceKind.ServiceKind* method), 522
- `__eq__()` (*rti.connexdds.ServiceRequest* method), 557
- `__eq__()` (*rti.connexdds.ServiceRequest.ContentFilteredTopic* method), 524
- `__eq__()` (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq* method), 525
- `__eq__()` (*rti.connexdds.ServiceRequest.DataReader* method), 528
- `__eq__()` (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 534
- `__eq__()` (*rti.connexdds.ServiceRequest.DataWriter* method), 536
- `__eq__()` (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 545
- `__eq__()` (*rti.connexdds.ServiceRequestId* method), 559
- `__eq__()` (*rti.connexdds.ServiceRequestId.ServiceRequestId* method), 559
- `__eq__()` (*rti.connexdds.ServiceRequestSeq* method), 561
- `__eq__()` (*rti.connexdds.ServiceRequestTimestampedSeq* method), 563
- `__eq__()` (*rti.connexdds.ServiceRequest.Topic* method), 552
- `__eq__()` (*rti.connexdds.ServiceRequest.TopicDescription* method), 553
- `__eq__()` (*rti.connexdds.ServiceRequest.TopicSeq* method), 554
- `__eq__()` (*rti.connexdds.StatusCondition* method), 565
- `__eq__()` (*rti.connexdds.StatusMask* method), 567
- `__eq__()` (*rti.connexdds.StreamKind* method), 570
- `__eq__()` (*rti.connexdds.StringPairSeq* method), 573
- `__eq__()` (*rti.connexdds.StringSeq* method), 575
- `__eq__()` (*rti.connexdds.StringType* method), 577
- `__eq__()` (*rti.connexdds.StructType* method), 577
- `__eq__()` (*rti.connexdds.Subscriber* method), 578
- `__eq__()` (*rti.connexdds.SubscriberQos* method), 580
- `__eq__()` (*rti.connexdds.SubscriberSeq* method), 583
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicData* method), 620
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic* method), 585
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 586
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader* method), 590
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 596
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* method), 598
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 608
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 623
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TimestampedSeq* method), 625
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicData.Topic* method), 615
- `__eq__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription* method), 615

- 615
- `__eq__` () (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 617
- `__eq__` () (*rti.connexdds.SyslogVerbosity* method), 629
- `__eq__` () (*rti.connexdds.SystemResourceLimits* method), 629
- `__eq__` () (*rti.connexdds.ThreadSettings* method), 630
- `__eq__` () (*rti.connexdds.ThreadSettingsCpuRotationKind* method), 632
- `__eq__` () (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* method), 631
- `__eq__` () (*rti.connexdds.ThreadSettingsKindMask* method), 633
- `__eq__` () (*rti.connexdds.Time* method), 636
- `__eq__` () (*rti.connexdds.TimeBasedFilter* method), 638
- `__eq__` () (*rti.connexdds.Topic* method), 638
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicData* method), 674
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic* method), 640
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 642
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicData.DataReader* method), 645
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq* method), 651
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicData.DataWriter* method), 653
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq* method), 662
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicDataSeq* method), 676
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq* method), 678
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicData.Topic* method), 669
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicData.TopicDescription* method), 670
- `__eq__` () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq* method), 671
- `__eq__` () (*rti.connexdds.TopicData* method), 681
- `__eq__` () (*rti.connexdds.TopicDescription* method), 681
- `__eq__` () (*rti.connexdds.TopicQos* method), 682
- `__eq__` () (*rti.connexdds.TopicQuery* method), 686
- `__eq__` () (*rti.connexdds.TopicQueryDispatch* method), 687
- `__eq__` () (*rti.connexdds.TopicQuerySelectionKind* method), 689
- `__eq__` () (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind* method), 688
- `__eq__` () (*rti.connexdds.TopicSeq* method), 690
- `__eq__` () (*rti.connexdds.TransportBuiltin* method), 692
- `__eq__` () (*rti.connexdds.TransportBuiltinMask* method), 693
- `__eq__` () (*rti.connexdds.TransportClassId* method), 698
- `__eq__` () (*rti.connexdds.TransportClassId.TransportClassId* method), 697
- `__eq__` () (*rti.connexdds.TransportInfoSeq* method), 700
- `__eq__` () (*rti.connexdds.TransportInfoVector* method), 701
- `__eq__` () (*rti.connexdds.TransportMulticast* method), 702
- `__eq__` () (*rti.connexdds.TransportMulticastKind* method), 704
- `__eq__` () (*rti.connexdds.TransportMulticastKind.TransportMulticastKind* method), 703
- `__eq__` () (*rti.connexdds.TransportMulticastMapping* method), 705
- `__eq__` () (*rti.connexdds.TransportMulticastMappingFunction* method), 705
- `__eq__` () (*rti.connexdds.TransportMulticastSeq* method), 706
- `__eq__` () (*rti.connexdds.TransportMulticastSettings* method), 708
- `__eq__` () (*rti.connexdds.TransportMulticastSettingsSeq* method), 709
- `__eq__` () (*rti.connexdds.TransportPriority* method), 711
- `__eq__` () (*rti.connexdds.TransportSelection* method), 712
- `__eq__` () (*rti.connexdds.TransportUnicast* method), 712
- `__eq__` () (*rti.connexdds.TransportUnicastSettings* method), 713
- `__eq__` () (*rti.connexdds.TransportUnicastSettingsSeq* method), 714
- `__eq__` () (*rti.connexdds.TypeConsistencyEnforcement* method), 716
- `__eq__` () (*rti.connexdds.TypeConsistencyEnforcementKind* method), 719
- `__eq__` () (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind* method), 718
- `__eq__` () (*rti.connexdds.TypeKind* method), 725
- `__eq__` () (*rti.connexdds.TypeKind.TypeKind* method), 723
- `__eq__` () (*rti.connexdds.TypeSupport* method), 725
- `__eq__` () (*rti.connexdds.Uint8Seq* method), 734
- `__eq__` () (*rti.connexdds.Uint8Type* method), 736
- `__eq__` () (*rti.connexdds.Uint16Seq* method), 726
- `__eq__` () (*rti.connexdds.Uint16Type* method), 728
- `__eq__` () (*rti.connexdds.Uint32Seq* method), 729
- `__eq__` () (*rti.connexdds.Uint32Type* method), 731
- `__eq__` () (*rti.connexdds.Uint64Seq* method), 732
- `__eq__` () (*rti.connexdds.Uint64Type* method), 734
- `__eq__` () (*rti.connexdds.UnidimensionalCollectionType* method), 737
- `__eq__` () (*rti.connexdds.UnionMember* method), 737
- `__eq__` () (*rti.connexdds.UnionMemberSeq* method), 738
- `__eq__` () (*rti.connexdds.UnionType* method), 740
- `__eq__` () (*rti.connexdds.UserData* method), 741
- `__eq__` () (*rti.connexdds.VendorId* method), 742
- `__eq__` () (*rti.connexdds.Verbosity* method), 744
- `__eq__` () (*rti.connexdds.Verbosity.Verbosity* method), 743
- `__eq__` () (*rti.connexdds.ViewState* method), 745
- `__eq__` () (*rti.connexdds.WaitSetProperty* method), 749
- `__eq__` () (*rti.connexdds.WcharSeq* method), 750
- `__eq__` () (*rti.connexdds.WcharType* method), 751
- `__eq__` () (*rti.connexdds.WireProtocol* method), 752

`__eq__()` (*rti.connexdds.WireProtocolAutoKind* method), 754
`__eq__()` (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* method), 753
`__eq__()` (*rti.connexdds.WriterDataLifecycle* method), 756
`__eq__()` (*rti.connexdds.WstringSeq* method), 757
`__eq__()` (*rti.connexdds.WstringType* method), 747
`__exit__()` (*rti.connexdds.CoherentAccess* method), 68
`__exit__()` (*rti.connexdds.CoherentSet* method), 68
`__exit__()` (*rti.connexdds.DataReader* method), 85
`__exit__()` (*rti.connexdds.DataReader.LoanedSamples* method), 83
`__exit__()` (*rti.connexdds.DataWriter* method), 114
`__exit__()` (*rti.connexdds.DomainParticipant* method), 157
`__exit__()` (*rti.connexdds.DynamicData.DataReader* method), 189
`__exit__()` (*rti.connexdds.DynamicData.DataWriter* method), 197
`__exit__()` (*rti.connexdds.DynamicData.LoanedSamples* method), 210
`__exit__()` (*rti.connexdds.DynamicData.ValidLoanedSamples* method), 219
`__exit__()` (*rti.connexdds.IReadCondition* method), 305
`__exit__()` (*rti.connexdds.LoanedDynamicData* method), 331
`__exit__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* method), 375
`__exit__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter* method), 383
`__exit__()` (*rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples* method), 395
`__exit__()` (*rti.connexdds.ParticipantBuiltinTopicData.ValidLoanedSamples* method), 403
`__exit__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader* method), 431
`__exit__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter* method), 439
`__exit__()` (*rti.connexdds.PublicationBuiltinTopicData.LoanedSamples* method), 452
`__exit__()` (*rti.connexdds.PublicationBuiltinTopicData.ValidLoanedSamples* method), 460
`__exit__()` (*rti.connexdds.ServiceRequest.DataReader* method), 529
`__exit__()` (*rti.connexdds.ServiceRequest.DataWriter* method), 536
`__exit__()` (*rti.connexdds.ServiceRequest.LoanedSamples* method), 548
`__exit__()` (*rti.connexdds.ServiceRequest.ValidLoanedSamples* method), 556
`__exit__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader* method), 590
`__exit__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* method), 598
`__exit__()` (*rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples* method), 610
`__exit__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ValidLoanedSamples* method), 618
`__exit__()` (*rti.connexdds.SuspendedPublication* method), 628
`__exit__()` (*rti.connexdds.ThreadContext* method), 630
`__exit__()` (*rti.connexdds.TopicBuiltinTopicData.DataReader* method), 645
`__exit__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriter* method), 653
`__exit__()` (*rti.connexdds.TopicBuiltinTopicData.LoanedSamples* method), 665
`__exit__()` (*rti.connexdds.TopicBuiltinTopicData.ValidLoanedSamples* method), 673
`__exit__()` (*rti.connexdds.TopicQuery* method), 686
`__float__()` (*rti.connexdds.Duration* method), 182
`__ge__()` (*rti.connexdds.AcknowledgmentKind* method), 36
`__ge__()` (*rti.connexdds.CdrPaddingKind* method), 62
`__ge__()` (*rti.connexdds.DataReaderInstanceRemovalKind* method), 93
`__ge__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind* method), 137
`__ge__()` (*rti.connexdds.DestinationOrderKind* method), 144
`__ge__()` (*rti.connexdds.DestinationOrderScopeKind* method), 146
`__ge__()` (*rti.connexdds.DurabilityKind* method), 180
`__ge__()` (*rti.connexdds.Duration* method), 182
`__ge__()` (*rti.connexdds.DynamicDataEncapsulationKind* method), 246
`__ge__()` (*rti.connexdds.ExtensibilityKind* method), 270
`__ge__()` (*rti.connexdds.FlowControllerSchedulingPolicy* method), 283
`__ge__()` (*rti.connexdds.Guid* method), 287
`__ge__()` (*rti.connexdds.HistoryKind* method), 290
`__ge__()` (*rti.connexdds.IgnoredEntityReplacementKind* method), 307
`__ge__()` (*rti.connexdds.LivelinessKind* method), 330
`__ge__()` (*rti.connexdds.LocatorKind* method), 337
`__ge__()` (*rti.connexdds.LogCategory* method), 343
`__ge__()` (*rti.connexdds.OwnershipKind* method), 368
`__ge__()` (*rti.connexdds.PresentationAccessScopeKind* method), 417
`__ge__()` (*rti.connexdds.PrintFormat* method), 420
`__ge__()` (*rti.connexdds.PrintFormatKind* method), 421
`__ge__()` (*rti.connexdds.PublishModeKind* method), 471

- `__ge__()` (*rti.connextdds.ReliabilityKind* method), 491
- `__ge__()` (*rti.connextdds.RemoteParticipantPurgeKind* method), 495
- `__ge__()` (*rti.connextdds.SequenceNumber* method), 519
- `__ge__()` (*rti.connextdds.ServiceKind* method), 523
- `__ge__()` (*rti.connextdds.ServiceRequestId* method), 560
- `__ge__()` (*rti.connextdds.ThreadSettingsCpuRotationKind* method), 632
- `__ge__()` (*rti.connextdds.Time* method), 636
- `__ge__()` (*rti.connextdds.TopicQuerySelectionKind* method), 689
- `__ge__()` (*rti.connextdds.TransportClassId* method), 698
- `__ge__()` (*rti.connextdds.TransportMulticastKind* method), 704
- `__ge__()` (*rti.connextdds.TypeConsistencyEnforcementKind* method), 719
- `__ge__()` (*rti.connextdds.TypeKind* method), 725
- `__ge__()` (*rti.connextdds.Verbosity* method), 744
- `__ge__()` (*rti.connextdds.WireProtocolAutoKind* method), 754
- `__getattr__()` (*rti.connextdds.ACTEnumMember* method), 31
- `__getitem__()` (*rti.connextdds.ACTEnumMember* method), 31
- `__getitem__()` (*rti.connextdds.ActivityContextMask* method), 38
- `__getitem__()` (*rti.connextdds.ACTMember* method), 32
- `__getitem__()` (*rti.connextdds.ACTUnionMember* method), 33
- `__getitem__()` (*rti.connextdds.AnyDataReaderSeq* method), 43
- `__getitem__()` (*rti.connextdds.AnyDataWriterSeq* method), 47
- `__getitem__()` (*rti.connextdds.AnyTopicSeq* method), 49
- `__getitem__()` (*rti.connextdds.BoolSeq* method), 54
- `__getitem__()` (*rti.connextdds.ByteVector* method), 60
- `__getitem__()` (*rti.connextdds.ChannelSettingsSeq* method), 64
- `__getitem__()` (*rti.connextdds.CharSeq* method), 66
- `__getitem__()` (*rti.connextdds.CompressionIdMask* method), 70
- `__getitem__()` (*rti.connextdds.ConditionSeq* method), 74
- `__getitem__()` (*rti.connextdds.ContentFilteredTopicSeq* method), 78
- `__getitem__()` (*rti.connextdds.CookieSeq* method), 80
- `__getitem__()` (*rti.connextdds.CookieVector* method), 82
- `__getitem__()` (*rti.connextdds.DataReader.LoanedSamples* method), 83
- `__getitem__()` (*rti.connextdds.DataReaderSeq* method), 107
- `__getitem__()` (*rti.connextdds.DataTag* method), 113
- `__getitem__()` (*rti.connextdds.DataWriterSeq* method), 138
- `__getitem__()` (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask* method), 151
- `__getitem__()` (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask* method), 154
- `__getitem__()` (*rti.connextdds.DomainParticipantSeq* method), 176
- `__getitem__()` (*rti.connextdds.DynamicData* method), 220
- `__getitem__()` (*rti.connextdds.DynamicData.ContentFilteredTopicSeq* method), 186
- `__getitem__()` (*rti.connextdds.DynamicData.DataReaderSeq* method), 195
- `__getitem__()` (*rti.connextdds.DynamicData.DataWriterSeq* method), 206
- `__getitem__()` (*rti.connextdds.DynamicData.LoanedSamples* method), 210
- `__getitem__()` (*rti.connextdds.DynamicDataSeq* method), 250
- `__getitem__()` (*rti.connextdds.DynamicData.SharedSamples* method), 214
- `__getitem__()` (*rti.connextdds.DynamicData.TimestampedSeq* method), 252
- `__getitem__()` (*rti.connextdds.DynamicData.TopicSeq* method), 217
- `__getitem__()` (*rti.connextdds.EndpointGroupSeq* method), 257
- `__getitem__()` (*rti.connextdds.EndpointGroupVector* method), 259
- `__getitem__()` (*rti.connextdds.EntitySeq* method), 261
- `__getitem__()` (*rti.connextdds.EnumMemberSeq* method), 264
- `__getitem__()` (*rti.connextdds.Float32Seq* method), 275
- `__getitem__()` (*rti.connextdds.Float64Seq* method), 277
- `__getitem__()` (*rti.connextdds.Float128Seq* method), 272
- `__getitem__()` (*rti.connextdds.Guid* method), 287
- `__getitem__()` (*rti.connextdds.IAnyDataReaderSeq* method), 291
- `__getitem__()` (*rti.connextdds.IAnyDataWriterSeq* method), 295
- `__getitem__()` (*rti.connextdds.IAnyTopicSeq* method), 297
- `__getitem__()` (*rti.connextdds.IConditionSeq* method), 300
- `__getitem__()` (*rti.connextdds.IEntitySeq* method), 303
- `__getitem__()` (*rti.connextdds.InstanceHandleSeq* method), 309
- `__getitem__()` (*rti.connextdds.InstanceState* method), 311
- `__getitem__()` (*rti.connextdds.Int8Seq* method), 323
- `__getitem__()` (*rti.connextdds.Int16Seq* method), 315
- `__getitem__()` (*rti.connextdds.Int32Seq* method), 317
- `__getitem__()` (*rti.connextdds.Int32Vector* method), 319
- `__getitem__()` (*rti.connextdds.Int64Seq* method), 321
- `__getitem__()` (*rti.connextdds.LocatorFilterElementSeq* method), 333
- `__getitem__()` (*rti.connextdds.LocatorSeq* method), 339
- `__getitem__()` (*rti.connextdds.LocatorVector* method), 340
- `__getitem__()` (*rti.connextdds.LongDouble* method), 345
- `__getitem__()` (*rti.connextdds.MemberSeq* method), 347
- `__getitem__()` (*rti.connextdds.MonitoringMetricSelectionSeq* method), 354
- `__getitem__()` (*rti.connextdds.MulticastMappingSeq* method), 358
- `__getitem__()` (*rti.connextdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 372

<code>__getitem__()</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq</i> method), 381	<code>__getitem__()</code> (<i>rti.connexdds.ServiceRequest.DataReaderSeq</i> method), 534
<code>__getitem__()</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq</i> method), 393	<code>__getitem__()</code> (<i>rti.connexdds.ServiceRequest.DataWriterSeq</i> method), 545
<code>__getitem__()</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples</i> method), 395	<code>__getitem__()</code> (<i>rti.connexdds.ServiceRequest.LoanedSamples</i> method), 548
<code>__getitem__()</code> (<i>rti.connexdds.ParticipantBuiltinTopicDataSeq</i> method), 407	<code>__getitem__()</code> (<i>rti.connexdds.ServiceRequestSeq</i> method), 561
<code>__getitem__()</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.SharedSamples</i> method), 399	<code>__getitem__()</code> (<i>rti.connexdds.ServiceRequest.SharedSamples</i> method), 551
<code>__getitem__()</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq</i> method), 409	<code>__getitem__()</code> (<i>rti.connexdds.ServiceRequestTimestampedSeq</i> method), 563
<code>__getitem__()</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.TopicSeq</i> method), 402	<code>__getitem__()</code> (<i>rti.connexdds.ServiceRequest.TopicSeq</i> method), 554
<code>__getitem__()</code> (<i>rti.connexdds.Property</i> method), 424	<code>__getitem__()</code> (<i>rti.connexdds.StatusMask</i> method), 567
<code>__getitem__()</code> (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq</i> method), 428	<code>__getitem__()</code> (<i>rti.connexdds.StreamKind</i> method), 570
<code>__getitem__()</code> (<i>rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq</i> method), 438	<code>__getitem__()</code> (<i>rti.connexdds.StringMap</i> method), 572
<code>__getitem__()</code> (<i>rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq</i> method), 449	<code>__getitem__()</code> (<i>rti.connexdds.StringPairSeq</i> method), 573
<code>__getitem__()</code> (<i>rti.connexdds.PublicationBuiltinTopicData.LoanedSamples</i> method), 452	<code>__getitem__()</code> (<i>rti.connexdds.StringSeq</i> method), 575
<code>__getitem__()</code> (<i>rti.connexdds.PublicationBuiltinTopicDataSeq</i> method), 464	<code>__getitem__()</code> (<i>rti.connexdds.SubscriberSeq</i> method), 583
<code>__getitem__()</code> (<i>rti.connexdds.PublicationBuiltinTopicData.SharedSamples</i> method), 456	<code>__getitem__()</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq</i> method), 586
<code>__getitem__()</code> (<i>rti.connexdds.PublicationBuiltinTopicData.TimestampedSeq</i> method), 467	<code>__getitem__()</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq</i> method), 596
<code>__getitem__()</code> (<i>rti.connexdds.PublicationBuiltinTopicData.TopicSeq</i> method), 458	<code>__getitem__()</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq</i> method), 608
<code>__getitem__()</code> (<i>rti.connexdds.PublisherSeq</i> method), 476	<code>__getitem__()</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples</i> method), 610
<code>__getitem__()</code> (<i>rti.connexdds.QosPolicyCountSeq</i> method), 479	<code>__getitem__()</code> (<i>rti.connexdds.SubscriptionBuiltinTopicDataSeq</i> method), 623
<code>__getitem__()</code> (<i>rti.connexdds.RtpsReservedPortKindMask</i> method), 501	<code>__getitem__()</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.SharedSamples</i> method), 614
<code>__getitem__()</code> (<i>rti.connexdds.SampleFlag</i> method), 505	<code>__getitem__()</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.TimestampedSeq</i> method), 625
<code>__getitem__()</code> (<i>rti.connexdds.SampleLostState</i> method), 511	<code>__getitem__()</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq</i> method), 617
<code>__getitem__()</code> (<i>rti.connexdds.SampleRejectedState</i> method), 514	<code>__getitem__()</code> (<i>rti.connexdds.ThreadSettingsKindMask</i> method), 633
<code>__getitem__()</code> (<i>rti.connexdds.SampleState</i> method), 517	<code>__getitem__()</code> (<i>rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq</i> method), 642
<code>__getitem__()</code> (<i>rti.connexdds.ServiceRequest.ContentFilteredTopicSeq</i> method), 525	<code>__getitem__()</code> (<i>rti.connexdds.TopicBuiltinTopicData.DataReaderSeq</i> method), 651
	<code>__getitem__()</code>

(*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method*), 662

`__getitem__()` (*rti.connexdds.TopicBuiltinTopicData.LoanedSamples method*), 665

`__getitem__()` (*rti.connexdds.TopicBuiltinTopicDataSeq method*), 676

`__getitem__()` (*rti.connexdds.TopicBuiltinTopicData.SharedSamples method*), 669

`__getitem__()` (*rti.connexdds.TopicBuiltinTopicData.TimestampedSeq method*), 679

`__getitem__()` (*rti.connexdds.TopicBuiltinTopicData.TopicSeq method*), 671

`__getitem__()` (*rti.connexdds.TopicSeq method*), 690

`__getitem__()` (*rti.connexdds.TransportBuiltinMask method*), 693

`__getitem__()` (*rti.connexdds.TransportInfoSeq method*), 700

`__getitem__()` (*rti.connexdds.TransportInfoVector method*), 701

`__getitem__()` (*rti.connexdds.TransportMulticastSeq method*), 706

`__getitem__()` (*rti.connexdds.TransportMulticastSettingsSeq method*), 709

`__getitem__()` (*rti.connexdds.TransportUnicastSettingsSeq method*), 714

`__getitem__()` (*rti.connexdds.TriggeredConditions method*), 716

`__getitem__()` (*rti.connexdds.Uint8Seq method*), 734

`__getitem__()` (*rti.connexdds.Uint16Seq method*), 727

`__getitem__()` (*rti.connexdds.Uint32Seq method*), 729

`__getitem__()` (*rti.connexdds.Uint64Seq method*), 732

`__getitem__()` (*rti.connexdds.UnionMemberSeq method*), 738

`__getitem__()` (*rti.connexdds.ViewState method*), 745

`__getitem__()` (*rti.connexdds.WcharSeq method*), 750

`__getitem__()` (*rti.connexdds.WstringSeq method*), 757

`__getstate__()` (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind method*), 36

`__getstate__()` (*rti.connexdds.CdrPaddingKind.CdrPaddingKind method*), 61

`__getstate__()` (*rti.connexdds.CharSeq method*), 66

`__getstate__()` (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 92

`__getstate__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 135

`__getstate__()` (*rti.connexdds.DestinationOrderKind.DestinationOrderKind method*), 143

`__getstate__()` (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 145

`__getstate__()` (*rti.connexdds.DurabilityKind.DurabilityKind method*), 180

`__getstate__()` (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 245

`__getstate__()` (*rti.connexdds.ExtensibilityKind.ExtensibilityKind method*), 269

`__getstate__()` (*rti.connexdds.Float32Seq method*), 275

`__getstate__()` (*rti.connexdds.Float64Seq method*), 277

`__getstate__()` (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 282

`__getstate__()` (*rti.connexdds.HistoryKind.HistoryKind method*), 289

`__getstate__()` (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 306

`__getstate__()` (*rti.connexdds.InstanceStateConsistencyKind method*), 314

`__getstate__()` (*rti.connexdds.Int8Seq method*), 324

`__getstate__()` (*rti.connexdds.Int16Seq method*), 315

`__getstate__()` (*rti.connexdds.Int32Seq method*), 317

`__getstate__()` (*rti.connexdds.Int64Seq method*), 321

`__getstate__()` (*rti.connexdds.LivelinessKind.LivelinessKind method*), 329

`__getstate__()` (*rti.connexdds LocatorKind.LocatorKind method*), 336

`__getstate__()` (*rti.connexdds.LogCategory.LogCategory method*), 342

`__getstate__()` (*rti.connexdds.OwnershipKind.OwnershipKind method*), 368

`__getstate__()` (*rti.connexdds.PersistentJournalKind method*), 412

`__getstate__()` (*rti.connexdds.PersistentSynchronizationKind method*), 414

`__getstate__()` (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 416

`__getstate__()` (*rti.connexdds.PrintFormatKind.PrintFormatKind method*), 421

`__getstate__()` (*rti.connexdds.PrintFormat.PrintFormat method*), 419

`__getstate__()` (*rti.connexdds.PublishModeKind.PublishModeKind method*), 471

`__getstate__()` (*rti.connexdds.ReliabilityKind.ReliabilityKind method*), 491

`__getstate__()` (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 494

- `__getstate__()` (*rti.connexdds.ServiceKind.ServiceKind method*), 522
- `__getstate__()` (*rti.connexdds.ServiceRequestId.ServiceRequestId method*), 559
- `__getstate__()` (*rti.connexdds.Syslog Verbosity method*), 629
- `__getstate__()` (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 631
- `__getstate__()` (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 689
- `__getstate__()` (*rti.connexdds.TransportClassId.TransportClassId method*), 697
- `__getstate__()` (*rti.connexdds.TransportMulticastKind.TransportMulticastKind method*), 703
- `__getstate__()` (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 718
- `__getstate__()` (*rti.connexdds.TypeKind.TypeKind method*), 723
- `__getstate__()` (*rti.connexdds.Uint8Seq method*), 734
- `__getstate__()` (*rti.connexdds.Uint16Seq method*), 727
- `__getstate__()` (*rti.connexdds.Uint32Seq method*), 729
- `__getstate__()` (*rti.connexdds.Uint64Seq method*), 732
- `__getstate__()` (*rti.connexdds.Verbosity.Verbosity method*), 743
- `__getstate__()` (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 753
- `__gt__()` (*rti.connexdds.AcknowledgmentKind method*), 37
- `__gt__()` (*rti.connexdds.CdrPaddingKind method*), 62
- `__gt__()` (*rti.connexdds.DataReaderInstanceRemovalKind method*), 93
- `__gt__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind method*), 137
- `__gt__()` (*rti.connexdds.DestinationOrderKind method*), 144
- `__gt__()` (*rti.connexdds.DestinationOrderScopeKind method*), 146
- `__gt__()` (*rti.connexdds.DurabilityKind method*), 180
- `__gt__()` (*rti.connexdds.Duration method*), 182
- `__gt__()` (*rti.connexdds.DynamicDataEncapsulationKind method*), 247
- `__gt__()` (*rti.connexdds.ExtensibilityKind method*), 270
- `__gt__()` (*rti.connexdds.FlowControllerSchedulingPolicy method*), 283
- `__gt__()` (*rti.connexdds.Guid method*), 287
- `__gt__()` (*rti.connexdds.HistoryKind method*), 290
- `__gt__()` (*rti.connexdds.IgnoredEntityReplacementKind method*), 307
- `__gt__()` (*rti.connexdds.LivelinessKind method*), 330
- `__gt__()` (*rti.connexdds.LocatorKind method*), 337
- `__gt__()` (*rti.connexdds.LogCategory method*), 343
- `__gt__()` (*rti.connexdds.OwnershipKind method*), 368
- `__gt__()` (*rti.connexdds.PresentationAccessScopeKind method*), 417
- `__gt__()` (*rti.connexdds.PrintFormat method*), 420
- `__gt__()` (*rti.connexdds.PrintFormatKind method*), 422
- `__gt__()` (*rti.connexdds.PublishModeKind method*), 471
- `__gt__()` (*rti.connexdds.ReliabilityKind method*), 491
- `__gt__()` (*rti.connexdds.RemoteParticipantPurgeKind method*), 495
- `__gt__()` (*rti.connexdds.SequenceNumber method*), 519
- `__gt__()` (*rti.connexdds.ServiceKind method*), 523
- `__gt__()` (*rti.connexdds.ServiceRequestId method*), 560
- `__gt__()` (*rti.connexdds.ThreadSettingsCpuRotationKind method*), 632
- `__gt__()` (*rti.connexdds.Time method*), 636
- `__gt__()` (*rti.connexdds.TopicQuerySelectionKind method*), 689
- `__gt__()` (*rti.connexdds.TransportClassId method*), 698
- `__gt__()` (*rti.connexdds.TransportMulticastKind method*), 704
- `__gt__()` (*rti.connexdds.TypeConsistencyEnforcementKind method*), 719
- `__gt__()` (*rti.connexdds.TypeKind method*), 725
- `__gt__()` (*rti.connexdds.Verbosity method*), 744
- `__gt__()` (*rti.connexdds.WireProtocolAutoKind method*), 754
- `__hash__` (*rti.connexdds.AcknowledgmentKind attribute*), 37
- `__hash__` (*rti.connexdds.ACTEnumMember attribute*), 31
- `__hash__` (*rti.connexdds.ActivityContextMask attribute*), 38
- `__hash__` (*rti.connexdds.ACTMember attribute*), 32
- `__hash__` (*rti.connexdds.ACTUnionMember attribute*), 33
- `__hash__` (*rti.connexdds.AliasType attribute*), 41
- `__hash__` (*rti.connexdds.AllocationSettings attribute*), 41
- `__hash__` (*rti.connexdds.AnyDataReaderSeq attribute*), 43
- `__hash__` (*rti.connexdds.AnyDataWriterSeq attribute*), 47
- `__hash__` (*rti.connexdds.AnyTopicSeq attribute*), 49
- `__hash__` (*rti.connexdds.ArrayType attribute*), 51
- `__hash__` (*rti.connexdds.AsynchronousPublisher attribute*), 51
- `__hash__` (*rti.connexdds.Availability attribute*), 52
- `__hash__` (*rti.connexdds.Batch attribute*), 53
- `__hash__` (*rti.connexdds.BoolSeq attribute*), 54
- `__hash__` (*rti.connexdds.BoolType attribute*), 56
- `__hash__` (*rti.connexdds.BuiltinTopicKey attribute*), 58
- `__hash__` (*rti.connexdds.BuiltinTopicReaderResourceLimits attribute*), 59
- `__hash__` (*rti.connexdds.ByteVector attribute*), 60
- `__hash__` (*rti.connexdds.CdrPaddingKind attribute*), 62
- `__hash__` (*rti.connexdds.ChannelSettings attribute*), 63
- `__hash__` (*rti.connexdds.ChannelSettingsSeq attribute*), 64
- `__hash__` (*rti.connexdds.CharSeq attribute*), 66
- `__hash__` (*rti.connexdds.CoherentSetInfo attribute*), 69
- `__hash__` (*rti.connexdds.CollectionType attribute*), 69
- `__hash__` (*rti.connexdds.CompressionIdMask attribute*), 70
- `__hash__` (*rti.connexdds.CompressionSettings attribute*), 73
- `__hash__` (*rti.connexdds.ConditionSeq attribute*), 74
- `__hash__` (*rti.connexdds.ContentFilteredTopic attribute*), 76
- `__hash__` (*rti.connexdds.ContentFilteredTopicSeq attribute*), 78
- `__hash__` (*rti.connexdds.ContentFilterProperty attribute*), 76
- `__hash__` (*rti.connexdds.Cookie attribute*), 79
- `__hash__` (*rti.connexdds.CookieSeq attribute*), 80

- `__hash__` (*rti.connexdds.CookieVector* attribute), 82
- `__hash__` (*rti.connexdds.Database* attribute), 140
- `__hash__` (*rti.connexdds.DataReader* attribute), 85
- `__hash__` (*rti.connexdds.DataReaderInstanceRemovalKind* attribute), 93
- `__hash__` (*rti.connexdds.DataReaderProtocol* attribute), 95
- `__hash__` (*rti.connexdds.DataReaderQos* attribute), 97
- `__hash__` (*rti.connexdds.DataReaderResourceLimits* attribute), 104
- `__hash__` (*rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings* attribute), 106
- `__hash__` (*rti.connexdds.DataReaderSeq* attribute), 108
- `__hash__` (*rti.connexdds.DataRepresentation* attribute), 109
- `__hash__` (*rti.connexdds.DataState* attribute), 110
- `__hash__` (*rti.connexdds.DataStateEx* attribute), 111
- `__hash__` (*rti.connexdds.DataTag* attribute), 113
- `__hash__` (*rti.connexdds.DataWriter* attribute), 114
- `__hash__` (*rti.connexdds.DataWriterProtocol* attribute), 122
- `__hash__` (*rti.connexdds.DataWriterQos* attribute), 124
- `__hash__` (*rti.connexdds.DataWriterResourceLimits* attribute), 132
- `__hash__` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind* attribute), 137
- `__hash__` (*rti.connexdds.DataWriterSeq* attribute), 138
- `__hash__` (*rti.connexdds.DataWriterShmemRefTransferModeSettings* attribute), 140
- `__hash__` (*rti.connexdds.DataWriterTransferMode* attribute), 140
- `__hash__` (*rti.connexdds.Deadline* attribute), 141
- `__hash__` (*rti.connexdds.DestinationOrder* attribute), 142
- `__hash__` (*rti.connexdds.DestinationOrderKind* attribute), 144
- `__hash__` (*rti.connexdds.DestinationOrderScopeKind* attribute), 146
- `__hash__` (*rti.connexdds.Discovery* attribute), 147
- `__hash__` (*rti.connexdds.DiscoveryConfig* attribute), 147
- `__hash__` (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* attribute), 151
- `__hash__` (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* attribute), 154
- `__hash__` (*rti.connexdds.DomainParticipant* attribute), 157
- `__hash__` (*rti.connexdds.DomainParticipantConfigParams* attribute), 163
- `__hash__` (*rti.connexdds.DomainParticipantFactoryQos* attribute), 164
- `__hash__` (*rti.connexdds.DomainParticipantQos* attribute), 166
- `__hash__` (*rti.connexdds.DomainParticipantResourceLimits* attribute), 171
- `__hash__` (*rti.connexdds.DomainParticipantSeq* attribute), 176
- `__hash__` (*rti.connexdds.Durability* attribute), 178
- `__hash__` (*rti.connexdds.DurabilityKind* attribute), 181
- `__hash__` (*rti.connexdds.DurabilityService* attribute), 181
- `__hash__` (*rti.connexdds.Duration* attribute), 182
- `__hash__` (*rti.connexdds.DynamicData* attribute), 221
- `__hash__` (*rti.connexdds.DynamicData.ContentFilteredTopic* attribute), 185
- `__hash__` (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* attribute), 186
- `__hash__` (*rti.connexdds.DynamicData.DataReader* attribute), 189
- `__hash__` (*rti.connexdds.DynamicData.DataReaderSeq* attribute), 195
- `__hash__` (*rti.connexdds.DynamicData.DataWriter* attribute), 197
- `__hash__` (*rti.connexdds.DynamicData.DataWriterSeq* attribute), 207
- `__hash__` (*rti.connexdds.DynamicDataEncapsulationKind* attribute), 247
- `__hash__` (*rti.connexdds.DynamicDataInfo* attribute), 247
- `__hash__` (*rti.connexdds.DynamicDataMemberInfo* attribute), 248
- `__hash__` (*rti.connexdds.DynamicDataProperty* attribute), 249
- `__hash__` (*rti.connexdds.DynamicDataSeq* attribute), 250
- `__hash__` (*rti.connexdds.DynamicDataTimestampedSeq* attribute), 252
- `__hash__` (*rti.connexdds.DynamicData.Topic* attribute), 214
- `__hash__` (*rti.connexdds.DynamicData.TopicDescription* attribute), 216
- `__hash__` (*rti.connexdds.DynamicData.TopicSeq* attribute), 217
- `__hash__` (*rti.connexdds.DynamicDataTypeSerializationProperty* attribute), 254
- `__hash__` (*rti.connexdds.DynamicType* attribute), 255
- `__hash__` (*rti.connexdds.DynamicTypePrintFormatProperty* attribute), 256
- `__hash__` (*rti.connexdds.EndpointGroupSeq* attribute), 257
- `__hash__` (*rti.connexdds.EndpointGroupVector* attribute), 259
- `__hash__` (*rti.connexdds.EntityFactory* attribute), 260
- `__hash__` (*rti.connexdds.EntityName* attribute), 260
- `__hash__` (*rti.connexdds.EntitySeq* attribute), 261
- `__hash__` (*rti.connexdds.EnumMember* attribute), 263
- `__hash__` (*rti.connexdds.EnumMemberSeq* attribute), 264
- `__hash__` (*rti.connexdds.EnumType* attribute), 266
- `__hash__` (*rti.connexdds.Event* attribute), 266
- `__hash__` (*rti.connexdds.ExclusiveArea* attribute), 268
- `__hash__` (*rti.connexdds.ExpressionProperty* attribute), 268
- `__hash__` (*rti.connexdds.ExtensibilityKind* attribute), 270
- `__hash__` (*rti.connexdds.FilterSampleInfo* attribute), 271
- `__hash__` (*rti.connexdds.Float32Seq* attribute), 275
- `__hash__` (*rti.connexdds.Float32Type* attribute), 277
- `__hash__` (*rti.connexdds.Float64Seq* attribute), 277
- `__hash__` (*rti.connexdds.Float64Type* attribute), 279
- `__hash__` (*rti.connexdds.Float128Seq* attribute), 272
- `__hash__` (*rti.connexdds.Float128Type* attribute), 274
- `__hash__` (*rti.connexdds.FlowController* attribute), 280
- `__hash__` (*rti.connexdds.FlowControllerProperty* attribute), 280
- `__hash__` (*rti.connexdds.FlowControllerSchedulingPolicy* attribute), 284
- `__hash__` (*rti.connexdds.FlowControllerTokenBucketProperty* attribute), 284

- __hash__ (rti.connexdds.GroupData attribute), 286
- __hash__ (rti.connexdds.GuardCondition attribute), 286
- __hash__ (rti.connexdds.Guid attribute), 287
- __hash__ (rti.connexdds.History attribute), 288
- __hash__ (rti.connexdds.HistoryKind attribute), 290
- __hash__ (rti.connexdds.IAnyDataReader attribute), 290
- __hash__ (rti.connexdds.IAnyDataReaderSeq attribute), 292
- __hash__ (rti.connexdds.IAnyDataWriter attribute), 293
- __hash__ (rti.connexdds.IAnyDataWriterSeq attribute), 295
- __hash__ (rti.connexdds.IAnyTopic attribute), 296
- __hash__ (rti.connexdds.IAnyTopicSeq attribute), 297
- __hash__ (rti.connexdds.ICondition attribute), 299
- __hash__ (rti.connexdds.IConditionSeq attribute), 300
- __hash__ (rti.connexdds.IEntity attribute), 302
- __hash__ (rti.connexdds.IEntitySeq attribute), 303
- __hash__ (rti.connexdds.IgnoredEntityReplacementKind attribute), 307
- __hash__ (rti.connexdds.InstanceHandleSeq attribute), 309
- __hash__ (rti.connexdds.InstanceState attribute), 311
- __hash__ (rti.connexdds.Int8Seq attribute), 324
- __hash__ (rti.connexdds.Int8Type attribute), 325
- __hash__ (rti.connexdds.Int16Seq attribute), 315
- __hash__ (rti.connexdds.Int16Type attribute), 317
- __hash__ (rti.connexdds.Int32Seq attribute), 317
- __hash__ (rti.connexdds.Int32Type attribute), 319
- __hash__ (rti.connexdds.Int32Vector attribute), 320
- __hash__ (rti.connexdds.Int64Seq attribute), 321
- __hash__ (rti.connexdds.Int64Type attribute), 323
- __hash__ (rti.connexdds.IReadCondition attribute), 305
- __hash__ (rti.connexdds.LatencyBudget attribute), 326
- __hash__ (rti.connexdds.Lifespan attribute), 326
- __hash__ (rti.connexdds.Liveliness attribute), 327
- __hash__ (rti.connexdds.LivelinessKind attribute), 330
- __hash__ (rti.connexdds.Locator attribute), 331
- __hash__ (rti.connexdds.LocatorFilter attribute), 332
- __hash__ (rti.connexdds.LocatorFilterElement attribute), 332
- __hash__ (rti.connexdds.LocatorFilterElementSeq attribute), 333
- __hash__ (rti.connexdds.LocatorKind attribute), 338
- __hash__ (rti.connexdds.LocatorSeq attribute), 339
- __hash__ (rti.connexdds.LocatorVector attribute), 340
- __hash__ (rti.connexdds.LogCategory attribute), 343
- __hash__ (rti.connexdds.LongDouble attribute), 345
- __hash__ (rti.connexdds.Member attribute), 346
- __hash__ (rti.connexdds.MemberSeq attribute), 347
- __hash__ (rti.connexdds.Monitoring attribute), 349
- __hash__ (rti.connexdds.MonitoringDedicatedParticipantSettings attribute), 350
- __hash__ (rti.connexdds.MonitoringDistributionSettings attribute), 350
- __hash__ (rti.connexdds.MonitoringEventDistributionSettings attribute), 351
- __hash__ (rti.connexdds.MonitoringLoggingDistributionSettings attribute), 351
- __hash__ (rti.connexdds.MonitoringLoggingForwardingSettings attribute), 352
- __hash__ (rti.connexdds.MonitoringMetricSelection attribute), 353
- __hash__ (rti.connexdds.MonitoringMetricSelectionSeq attribute), 354
- __hash__ (rti.connexdds.MonitoringPeriodicDistributionSettings attribute), 356
- __hash__ (rti.connexdds.MonitoringTelemetryData attribute), 356
- __hash__ (rti.connexdds.MulticastMapping attribute), 357
- __hash__ (rti.connexdds.MulticastMappingSeq attribute), 359
- __hash__ (rti.connexdds.MultiChannel attribute), 357
- __hash__ (rti.connexdds.Ownership attribute), 367
- __hash__ (rti.connexdds.OwnershipKind attribute), 368
- __hash__ (rti.connexdds.OwnershipStrength attribute), 369
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData attribute), 405
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic attribute), 370
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq attribute), 372
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData.DataReader attribute), 375
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq attribute), 381
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData.DataWriter attribute), 383
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq attribute), 393
- __hash__ (rti.connexdds.ParticipantBuiltinTopicDataSeq attribute), 407
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq attribute), 409
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData.Topic attribute), 400
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData.TopicDescription attribute), 401
- __hash__ (rti.connexdds.ParticipantBuiltinTopicData.TopicSeq attribute), 402
- __hash__ (rti.connexdds.Partition attribute), 411
- __hash__ (rti.connexdds.PersistentStorageSettings attribute), 413
- __hash__ (rti.connexdds.Presentation attribute), 415
- __hash__ (rti.connexdds.PresentationAccessScopeKind attribute), 417
- __hash__ (rti.connexdds.PrintFormat attribute), 420
- __hash__ (rti.connexdds.PrintFormatKind attribute), 422
- __hash__ (rti.connexdds.ProductVersion attribute), 423
- __hash__ (rti.connexdds.Property attribute), 424
- __hash__ (rti.connexdds.ProtocolVersion attribute), 425

<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicData</i> attribute), 461	<code>__hash__</code> (<i>rti.connexdds.SampleFlag</i> attribute), 505
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic</i> attribute), 426	<code>__hash__</code> (<i>rti.connexdds.SampleIdentity</i> attribute), 507
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq</i> attribute), 428	<code>__hash__</code> (<i>rti.connexdds.SampleInfo</i> attribute), 508
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.DataReader</i> attribute), 431	<code>__hash__</code> (<i>rti.connexdds.SampleLostState</i> attribute), 511
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq</i> attribute), 438	<code>__hash__</code> (<i>rti.connexdds.SampleRejectedState</i> attribute), 514
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.DataWriter</i> attribute), 440	<code>__hash__</code> (<i>rti.connexdds.SampleState</i> attribute), 517
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq</i> attribute), 449	<code>__hash__</code> (<i>rti.connexdds.SequenceNumber</i> attribute), 519
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicDataSeq</i> attribute), 464	<code>__hash__</code> (<i>rti.connexdds.SequenceType</i> attribute), 520
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq</i> attribute), 467	<code>__hash__</code> (<i>rti.connexdds.Service</i> attribute), 521
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.Topic</i> attribute), 456	<code>__hash__</code> (<i>rti.connexdds.ServiceKind</i> attribute), 523
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.TopicDescription</i> attribute), 457	<code>__hash__</code> (<i>rti.connexdds.ServiceRequest</i> attribute), 557
<code>__hash__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.TopicSeq</i> attribute), 458	<code>__hash__</code> (<i>rti.connexdds.ServiceRequest.ContentFilteredTopic</i> attribute), 524
<code>__hash__</code> (<i>rti.connexdds.Publisher</i> attribute), 472	<code>__hash__</code> (<i>rti.connexdds.ServiceRequest.ContentFilteredTopicSeq</i> attribute), 526
<code>__hash__</code> (<i>rti.connexdds.PublisherQos</i> attribute), 474	<code>__hash__</code> (<i>rti.connexdds.ServiceRequest.DataReader</i> attribute), 529
<code>__hash__</code> (<i>rti.connexdds.PublisherSeq</i> attribute), 476	<code>__hash__</code> (<i>rti.connexdds.ServiceRequest.DataReaderSeq</i> attribute), 534
<code>__hash__</code> (<i>rti.connexdds.PublishMode</i> attribute), 469	<code>__hash__</code> (<i>rti.connexdds.ServiceRequest.DataWriter</i> attribute), 536
<code>__hash__</code> (<i>rti.connexdds.PublishModeKind</i> attribute), 471	<code>__hash__</code> (<i>rti.connexdds.ServiceRequest.DataWriterSeq</i> attribute), 546
<code>__hash__</code> (<i>rti.connexdds.QosPolicyCount</i> attribute), 478	<code>__hash__</code> (<i>rti.connexdds.ServiceRequestId</i> attribute), 560
<code>__hash__</code> (<i>rti.connexdds.QosPolicyCountSeq</i> attribute), 479	<code>__hash__</code> (<i>rti.connexdds.ServiceRequestSeq</i> attribute), 561
<code>__hash__</code> (<i>rti.connexdds.QosPrintFormat</i> attribute), 480	<code>__hash__</code> (<i>rti.connexdds.ServiceRequestTimestampedSeq</i> attribute), 563
<code>__hash__</code> (<i>rti.connexdds.QosProvider</i> attribute), 481	<code>__hash__</code> (<i>rti.connexdds.ServiceRequest.Topic</i> attribute), 552
<code>__hash__</code> (<i>rti.connexdds.QosProviderParams</i> attribute), 484	<code>__hash__</code> (<i>rti.connexdds.ServiceRequest.TopicDescription</i> attribute), 553
<code>__hash__</code> (<i>rti.connexdds.Query</i> attribute), 485	<code>__hash__</code> (<i>rti.connexdds.ServiceRequest.TopicSeq</i> attribute), 554
<code>__hash__</code> (<i>rti.connexdds.QueryCondition</i> attribute), 486	<code>__hash__</code> (<i>rti.connexdds.StatusCondition</i> attribute), 565
<code>__hash__</code> (<i>rti.connexdds.ReaderDataLifecycle</i> attribute), 488	<code>__hash__</code> (<i>rti.connexdds.StatusMask</i> attribute), 567
<code>__hash__</code> (<i>rti.connexdds.ReceiverPool</i> attribute), 489	<code>__hash__</code> (<i>rti.connexdds.StreamKind</i> attribute), 570
<code>__hash__</code> (<i>rti.connexdds.Reliability</i> attribute), 489	<code>__hash__</code> (<i>rti.connexdds.StringPairSeq</i> attribute), 573
<code>__hash__</code> (<i>rti.connexdds.ReliabilityKind</i> attribute), 491	<code>__hash__</code> (<i>rti.connexdds.StringSeq</i> attribute), 575
<code>__hash__</code> (<i>rti.connexdds.RemoteParticipantPurgeKind</i> attribute), 495	<code>__hash__</code> (<i>rti.connexdds.StringType</i> attribute), 577
<code>__hash__</code> (<i>rti.connexdds.ResourceLimits</i> attribute), 496	<code>__hash__</code> (<i>rti.connexdds.StructType</i> attribute), 577
<code>__hash__</code> (<i>rti.connexdds.RtpsReliableReaderProtocol</i> attribute), 497	<code>__hash__</code> (<i>rti.connexdds.Subscriber</i> attribute), 578
<code>__hash__</code> (<i>rti.connexdds.RtpsReliableWriterProtocol</i> attribute), 498	<code>__hash__</code> (<i>rti.connexdds.SubscriberQos</i> attribute), 580
<code>__hash__</code> (<i>rti.connexdds.RtpsReservedPortKindMask</i> attribute), 501	<code>__hash__</code> (<i>rti.connexdds.SubscriberSeq</i> attribute), 583
<code>__hash__</code> (<i>rti.connexdds.RtpsWellKnownPorts</i> attribute), 504	<code>__hash__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData</i> attribute), 620
	<code>__hash__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic</i> attribute), 585
	<code>__hash__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq</i> attribute), 586
	<code>__hash__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataReader</i> attribute), 590
	<code>__hash__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq</i> attribute), 596
	<code>__hash__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataWriter</i> attribute),

- 598
- __hash__ (rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq attribute), 608
- __hash__ (rti.connexdds.SubscriptionBuiltinTopicDataSeq attribute), 623
- __hash__ (rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq attribute), 625
- __hash__ (rti.connexdds.SubscriptionBuiltinTopicData.Topic attribute), 615
- __hash__ (rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription attribute), 616
- __hash__ (rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq attribute), 617
- __hash__ (rti.connexdds.SystemResourceLimits attribute), 629
- __hash__ (rti.connexdds.ThreadSettings attribute), 630
- __hash__ (rti.connexdds.ThreadSettingsCpuRotationKind attribute), 632
- __hash__ (rti.connexdds.ThreadSettingsKindMask attribute), 634
- __hash__ (rti.connexdds.Time attribute), 636
- __hash__ (rti.connexdds.TimeBasedFilter attribute), 638
- __hash__ (rti.connexdds.Topic attribute), 638
- __hash__ (rti.connexdds.TopicBuiltinTopicData attribute), 674
- __hash__ (rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic attribute), 640
- __hash__ (rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq attribute), 642
- __hash__ (rti.connexdds.TopicBuiltinTopicData.DataReader attribute), 645
- __hash__ (rti.connexdds.TopicBuiltinTopicData.DataReaderSeq attribute), 651
- __hash__ (rti.connexdds.TopicBuiltinTopicData.DataWriter attribute), 653
- __hash__ (rti.connexdds.TopicBuiltinTopicData.DataWriterSeq attribute), 663
- __hash__ (rti.connexdds.TopicBuiltinTopicDataSeq attribute), 676
- __hash__ (rti.connexdds.TopicBuiltinTopicDataTimestampedSeq attribute), 679
- __hash__ (rti.connexdds.TopicBuiltinTopicData.Topic attribute), 669
- __hash__ (rti.connexdds.TopicBuiltinTopicData.TopicDescription attribute), 670
- __hash__ (rti.connexdds.TopicBuiltinTopicData.TopicSeq attribute), 671
- __hash__ (rti.connexdds.TopicData attribute), 681
- __hash__ (rti.connexdds.TopicDescription attribute), 681
- __hash__ (rti.connexdds.TopicQos attribute), 682
- __hash__ (rti.connexdds.TopicQuery attribute), 686
- __hash__ (rti.connexdds.TopicQueryDispatch attribute), 687
- __hash__ (rti.connexdds.TopicQuerySelectionKind attribute), 689
- __hash__ (rti.connexdds.TopicSeq attribute), 691
- __hash__ (rti.connexdds.TransportBuiltin attribute), 692
- __hash__ (rti.connexdds.TransportBuiltinMask attribute), 693
- __hash__ (rti.connexdds.TransportClassId attribute), 698
- __hash__ (rti.connexdds.TransportInfoSeq attribute), 700
- __hash__ (rti.connexdds.TransportInfoVector attribute), 701
- __hash__ (rti.connexdds.TransportMulticast attribute), 702
- __hash__ (rti.connexdds.TransportMulticastKind attribute), 704
- __hash__ (rti.connexdds.TransportMulticastMapping attribute), 705
- __hash__ (rti.connexdds.TransportMulticastMappingFunction attribute), 705
- __hash__ (rti.connexdds.TransportMulticastSeq attribute), 706
- __hash__ (rti.connexdds.TransportMulticastSettings attribute), 708
- __hash__ (rti.connexdds.TransportMulticastSettingsSeq attribute), 709
- __hash__ (rti.connexdds.TransportPriority attribute), 711
- __hash__ (rti.connexdds.TransportSelection attribute), 712
- __hash__ (rti.connexdds.TransportUnicast attribute), 712
- __hash__ (rti.connexdds.TransportUnicastSettings attribute), 713
- __hash__ (rti.connexdds.TransportUnicastSettingsSeq attribute), 714
- __hash__ (rti.connexdds.TypeConsistencyEnforcement attribute), 716
- __hash__ (rti.connexdds.TypeConsistencyEnforcementKind attribute), 719
- __hash__ (rti.connexdds.TypeKind attribute), 725
- __hash__ (rti.connexdds.TypeSupport attribute), 726
- __hash__ (rti.connexdds.Uint8Seq attribute), 735
- __hash__ (rti.connexdds.Uint8Type attribute), 736
- __hash__ (rti.connexdds.Uint16Seq attribute), 727
- __hash__ (rti.connexdds.Uint16Type attribute), 729
- __hash__ (rti.connexdds.Uint32Seq attribute), 729
- __hash__ (rti.connexdds.Uint32Type attribute), 731
- __hash__ (rti.connexdds.Uint64Seq attribute), 732
- __hash__ (rti.connexdds.Uint64Type attribute), 734
- __hash__ (rti.connexdds.UnidimensionalCollectionType attribute), 737
- __hash__ (rti.connexdds.UnionMember attribute), 737
- __hash__ (rti.connexdds.UnionMemberSeq attribute), 738
- __hash__ (rti.connexdds.UnionType attribute), 740
- __hash__ (rti.connexdds.UserData attribute), 741
- __hash__ (rti.connexdds.VendorId attribute), 742
- __hash__ (rti.connexdds.Verbosity attribute), 744
- __hash__ (rti.connexdds.ViewState attribute), 745
- __hash__ (rti.connexdds.WaitSetProperty attribute), 749
- __hash__ (rti.connexdds.WcharSeq attribute), 750
- __hash__ (rti.connexdds.WcharType attribute), 751
- __hash__ (rti.connexdds.WireProtocol attribute), 752
- __hash__ (rti.connexdds.WireProtocolAutoKind attribute), 754
- __hash__ (rti.connexdds.WriterDataLifecycle attribute), 756
- __hash__ (rti.connexdds.WstringSeq attribute), 757
- __hash__ (rti.connexdds.WstringType attribute), 747

- `__hash__` () (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind* method), 36
- `__hash__` () (*rti.connexdds.CdrPaddingKind.CdrPaddingKind* method), 61
- `__hash__` () (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind* method), 92
- `__hash__` () (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind* method), 136
- `__hash__` () (*rti.connexdds.DestinationOrderKind.DestinationOrderKind* method), 143
- `__hash__` () (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind* method), 145
- `__hash__` () (*rti.connexdds.DurabilityKind.DurabilityKind* method), 180
- `__hash__` () (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind* method), 245
- `__hash__` () (*rti.connexdds.ExtensibilityKind.ExtensibilityKind* method), 269
- `__hash__` () (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy* method), 282
- `__hash__` () (*rti.connexdds.HistoryKind.HistoryKind* method), 289
- `__hash__` () (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind* method), 306
- `__hash__` () (*rti.connexdds.InstanceHandle* method), 308
- `__hash__` () (*rti.connexdds.InstanceStateConsistencyKind* method), 314
- `__hash__` () (*rti.connexdds.LivelinessKind.LivelinessKind* method), 329
- `__hash__` () (*rti.connexdds.LocatorKind.LocatorKind* method), 336
- `__hash__` () (*rti.connexdds.LogCategory.LogCategory* method), 342
- `__hash__` () (*rti.connexdds.OwnershipKind.OwnershipKind* method), 368
- `__hash__` () (*rti.connexdds.PersistentJournalKind* method), 412
- `__hash__` () (*rti.connexdds.PersistentSynchronizationKind* method), 414
- `__hash__` () (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind* method), 416
- `__hash__` () (*rti.connexdds.PrintFormatKind.PrintFormatKind* method), 421
- `__hash__` () (*rti.connexdds.PrintFormat.PrintFormat* method), 419
- `__hash__` () (*rti.connexdds.PublishModeKind.PublishModeKind* method), 471
- `__hash__` () (*rti.connexdds.ReliabilityKind.ReliabilityKind* method), 491
- `__hash__` () (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* method), 494
- `__hash__` () (*rti.connexdds.ServiceKind.ServiceKind* method), 522
- `__hash__` () (*rti.connexdds.ServiceRequestId.ServiceRequestId* method), 559
- `__hash__` () (*rti.connexdds.SyslogVerbosity* method), 629
- `__hash__` () (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* method), 631
- `__hash__` () (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind* method), 689
- `__hash__` () (*rti.connexdds.TransportClassId.TransportClassId* method), 697
- `__hash__` () (*rti.connexdds.TransportMulticastKind.TransportMulticastKind* method), 703
- `__hash__` () (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind* method), 718
- `__hash__` () (*rti.connexdds.TypeKind.TypeKind* method), 723
- `__hash__` () (*rti.connexdds.Verbosity.Verbosity* method), 743
- `__hash__` () (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* method), 753
- `__iadd__` () (*rti.connexdds.AnyDataReaderSeq* method), 43
- `__iadd__` () (*rti.connexdds.AnyDataWriterSeq* method), 47
- `__iadd__` () (*rti.connexdds.AnyTopicSeq* method), 49
- `__iadd__` () (*rti.connexdds.BoolSeq* method), 54
- `__iadd__` () (*rti.connexdds.ChannelSettingsSeq* method), 64
- `__iadd__` () (*rti.connexdds.CharSeq* method), 66
- `__iadd__` () (*rti.connexdds.ConditionSeq* method), 74
- `__iadd__` () (*rti.connexdds.ContentFilteredTopicSeq* method), 78
- `__iadd__` () (*rti.connexdds.CookieSeq* method), 80
- `__iadd__` () (*rti.connexdds.DataReaderSeq* method), 108
- `__iadd__` () (*rti.connexdds.DataWriterSeq* method), 138
- `__iadd__` () (*rti.connexdds.DomainParticipantSeq* method), 176
- `__iadd__` () (*rti.connexdds.Duration* method), 182
- `__iadd__` () (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 186
- `__iadd__` () (*rti.connexdds.DynamicData.DataReaderSeq* method), 195
- `__iadd__` () (*rti.connexdds.DynamicData.DataWriterSeq* method), 207
- `__iadd__` () (*rti.connexdds.DynamicDataSeq* method), 250
- `__iadd__` () (*rti.connexdds.DynamicData.TimestampedSeq* method), 252
- `__iadd__` () (*rti.connexdds.DynamicData.TopicSeq* method), 217
- `__iadd__` () (*rti.connexdds.EndpointGroupSeq* method), 257
- `__iadd__` () (*rti.connexdds.EntitySeq* method), 261

- `__iadd__()` (*rti.connexdds.EnumMemberSeq* method), 264
- `__iadd__()` (*rti.connexdds.Float32Seq* method), 275
- `__iadd__()` (*rti.connexdds.Float64Seq* method), 277
- `__iadd__()` (*rti.connexdds.Float128Seq* method), 272
- `__iadd__()` (*rti.connexdds.IAnyDataReaderSeq* method), 292
- `__iadd__()` (*rti.connexdds.IAnyDataWriterSeq* method), 295
- `__iadd__()` (*rti.connexdds.IAnyTopicSeq* method), 297
- `__iadd__()` (*rti.connexdds.IConditionSeq* method), 300
- `__iadd__()` (*rti.connexdds.IEntitySeq* method), 303
- `__iadd__()` (*rti.connexdds.InstanceHandleSeq* method), 309
- `__iadd__()` (*rti.connexdds.Int8Seq* method), 324
- `__iadd__()` (*rti.connexdds.Int16Seq* method), 315
- `__iadd__()` (*rti.connexdds.Int32Seq* method), 317
- `__iadd__()` (*rti.connexdds.Int64Seq* method), 321
- `__iadd__()` (*rti.connexdds.LocatorFilterElementSeq* method), 333
- `__iadd__()` (*rti.connexdds.LocatorSeq* method), 339
- `__iadd__()` (*rti.connexdds.MemberSeq* method), 347
- `__iadd__()` (*rti.connexdds.MonitoringMetricSelectionSeq* method), 354
- `__iadd__()` (*rti.connexdds.MulticastMappingSeq* method), 359
- `__iadd__()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 372
- `__iadd__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 381
- `__iadd__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 393
- `__iadd__()` (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 407
- `__iadd__()` (*rti.connexdds.ParticipantBuiltinTopicData-TimestampedSeq* method), 409
- `__iadd__()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 402
- `__iadd__()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 428
- `__iadd__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 438
- `__iadd__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 449
- `__iadd__()` (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 464
- `__iadd__()` (*rti.connexdds.PublicationBuiltinTopicData-TimestampedSeq* method), 467
- `__iadd__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 458
- `__iadd__()` (*rti.connexdds.PublisherSeq* method), 477
- `__iadd__()` (*rti.connexdds.QosPolicyCountSeq* method), 479
- `__iadd__()` (*rti.connexdds.SequenceNumber* method), 519
- `__iadd__()` (*rti.connexdds.ServiceRequest.ContentFiltered-TopicSeq* method), 526
- `__iadd__()` (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 535
- `__iadd__()` (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 546
- `__iadd__()` (*rti.connexdds.ServiceRequestSeq* method), 561
- `__iadd__()` (*rti.connexdds.ServiceRequestTimestampedSeq* method), 563
- `__iadd__()` (*rti.connexdds.ServiceRequest.TopicSeq* method), 554
- `__iadd__()` (*rti.connexdds.StringPairSeq* method), 573
- `__iadd__()` (*rti.connexdds.StringSeq* method), 575
- `__iadd__()` (*rti.connexdds.SubscriberSeq* method), 583
- `__iadd__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 586
- `__iadd__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 596
- `__iadd__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 608
- `__iadd__()` (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 623
- `__iadd__()` (*rti.connexdds.SubscriptionBuiltinTopicData-TimestampedSeq* method), 625
- `__iadd__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 617
- `__iadd__()` (*rti.connexdds.Time* method), 636
- `__iadd__()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 642
- `__iadd__()` (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq* method), 651
- `__iadd__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq* method), 663
- `__iadd__()` (*rti.connexdds.TopicBuiltinTopicDataSeq* method), 676
- `__iadd__()` (*rti.connexdds.TopicBuiltinTopicData-TimestampedSeq* method), 679
- `__iadd__()` (*rti.connexdds.TopicBuiltinTopicData.TopicSeq* method), 671
- `__iadd__()` (*rti.connexdds.TopicSeq* method), 691
- `__iadd__()` (*rti.connexdds.TransportInfoSeq* method), 700
- `__iadd__()` (*rti.connexdds.TransportMulticastSeq* method), 707
- `__iadd__()` (*rti.connexdds.TransportMulticastSettingsSeq* method), 709
- `__iadd__()` (*rti.connexdds.TransportUnicastSettingsSeq* method), 714
- `__iadd__()` (*rti.connexdds.Uint8Seq* method), 735
- `__iadd__()` (*rti.connexdds.Uint16Seq* method), 727

- `__iadd__` () (*rti.connexdds.Uint32Seq* method), 729
- `__iadd__` () (*rti.connexdds.Uint64Seq* method), 732
- `__iadd__` () (*rti.connexdds.UnionMemberSeq* method), 738
- `__iadd__` () (*rti.connexdds.WaitSet* method), 747
- `__iadd__` () (*rti.connexdds.WcharSeq* method), 750
- `__iadd__` () (*rti.connexdds.WstringSeq* method), 757
- `__iand__` () (*rti.connexdds.ActivityContextMask* method), 38
- `__iand__` () (*rti.connexdds.CompressionIdMask* method), 70
- `__iand__` () (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 151
- `__iand__` () (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 154
- `__iand__` () (*rti.connexdds.InstanceState* method), 311
- `__iand__` () (*rti.connexdds.RtpsReservedPortKindMask* method), 501
- `__iand__` () (*rti.connexdds.SampleFlag* method), 505
- `__iand__` () (*rti.connexdds.SampleLostState* method), 511
- `__iand__` () (*rti.connexdds.SampleRejectedState* method), 514
- `__iand__` () (*rti.connexdds.SampleState* method), 517
- `__iand__` () (*rti.connexdds.StatusMask* method), 567
- `__iand__` () (*rti.connexdds.StreamKind* method), 570
- `__iand__` () (*rti.connexdds.ThreadSettingsKindMask* method), 634
- `__iand__` () (*rti.connexdds.TransportBuiltinMask* method), 693
- `__iand__` () (*rti.connexdds.ViewState* method), 745
- `__ilshift__` () (*rti.connexdds.ActivityContextMask* method), 38
- `__ilshift__` () (*rti.connexdds.CompressionIdMask* method), 70
- `__ilshift__` () (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 151
- `__ilshift__` () (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 154
- `__ilshift__` () (*rti.connexdds.InstanceState* method), 311
- `__ilshift__` () (*rti.connexdds.RtpsReservedPortKindMask* method), 501
- `__ilshift__` () (*rti.connexdds.SampleFlag* method), 505
- `__ilshift__` () (*rti.connexdds.SampleLostState* method), 511
- `__ilshift__` () (*rti.connexdds.SampleRejectedState* method), 514
- `__ilshift__` () (*rti.connexdds.SampleState* method), 517
- `__ilshift__` () (*rti.connexdds.StatusMask* method), 567
- `__ilshift__` () (*rti.connexdds.StreamKind* method), 570
- `__ilshift__` () (*rti.connexdds.ThreadSettingsKindMask* method), 634
- `__ilshift__` () (*rti.connexdds.TransportBuiltinMask* method), 693
- `__ilshift__` () (*rti.connexdds.ViewState* method), 745
- `__imul__` () (*rti.connexdds.AnyDataReaderSeq* method), 43
- `__imul__` () (*rti.connexdds.AnyDataWriterSeq* method), 47
- `__imul__` () (*rti.connexdds.AnyTopicSeq* method), 49
- `__imul__` () (*rti.connexdds.BoolSeq* method), 54
- `__imul__` () (*rti.connexdds.ChannelSettingsSeq* method), 64
- `__imul__` () (*rti.connexdds.CharSeq* method), 66
- `__imul__` () (*rti.connexdds.ConditionSeq* method), 74
- `__imul__` () (*rti.connexdds.ContentFilteredTopicSeq* method), 78
- `__imul__` () (*rti.connexdds.CookieSeq* method), 81
- `__imul__` () (*rti.connexdds.DataReaderSeq* method), 108
- `__imul__` () (*rti.connexdds.DataWriterSeq* method), 138
- `__imul__` () (*rti.connexdds.DomainParticipantSeq* method), 176
- `__imul__` () (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 187
- `__imul__` () (*rti.connexdds.DynamicData.DataReaderSeq* method), 196
- `__imul__` () (*rti.connexdds.DynamicData.DataWriterSeq* method), 207
- `__imul__` () (*rti.connexdds.DynamicDataSeq* method), 250
- `__imul__` () (*rti.connexdds.DynamicDataTimestampedSeq* method), 252
- `__imul__` () (*rti.connexdds.DynamicData.TopicSeq* method), 217
- `__imul__` () (*rti.connexdds.EndpointGroupSeq* method), 257
- `__imul__` () (*rti.connexdds.EntitySeq* method), 261
- `__imul__` () (*rti.connexdds.EnumMemberSeq* method), 264
- `__imul__` () (*rti.connexdds.Float32Seq* method), 275
- `__imul__` () (*rti.connexdds.Float64Seq* method), 277
- `__imul__` () (*rti.connexdds.Float128Seq* method), 272
- `__imul__` () (*rti.connexdds.IAnyDataReaderSeq* method), 292
- `__imul__` () (*rti.connexdds.IAnyDataWriterSeq* method), 295
- `__imul__` () (*rti.connexdds.IAnyTopicSeq* method), 297
- `__imul__` () (*rti.connexdds.IConditionSeq* method), 300
- `__imul__` () (*rti.connexdds.IEntitySeq* method), 303
- `__imul__` () (*rti.connexdds.InstanceHandleSeq* method), 309
- `__imul__` () (*rti.connexdds.Int16Seq* method), 324
- `__imul__` () (*rti.connexdds.Int16Seq* method), 315
- `__imul__` () (*rti.connexdds.Int32Seq* method), 317
- `__imul__` () (*rti.connexdds.Int64Seq* method), 321
- `__imul__` () (*rti.connexdds.LocatorFilterElementSeq* method), 333
- `__imul__` () (*rti.connexdds.LocatorSeq* method), 339
- `__imul__` () (*rti.connexdds.MemberSeq* method), 347
- `__imul__` () (*rti.connexdds.MonitoringMetricSelectionSeq* method), 354
- `__imul__` () (*rti.connexdds.MulticastMappingSeq* method), 359
- `__imul__` () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 372
- `__imul__` () (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 381
- `__imul__` () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 393
- `__imul__` () (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 407
- `__imul__` () (*rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq* method), 409

- `__imul__()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq method*), 402
- `__imul__()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 428
- `__imul__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq method*), 438
- `__imul__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq method*), 449
- `__imul__()` (*rti.connexdds.PublicationBuiltinTopicDataSeq method*), 464
- `__imul__()` (*rti.connexdds.PublicationBuiltinTopicData-TimestampedSeq method*), 467
- `__imul__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq method*), 458
- `__imul__()` (*rti.connexdds.PublisherSeq method*), 477
- `__imul__()` (*rti.connexdds.QosPolicyCountSeq method*), 479
- `__imul__()` (*rti.connexdds.ServiceRequest.ContentFiltered-TopicSeq method*), 526
- `__imul__()` (*rti.connexdds.ServiceRequest.DataReaderSeq method*), 535
- `__imul__()` (*rti.connexdds.ServiceRequest.DataWriterSeq method*), 546
- `__imul__()` (*rti.connexdds.ServiceRequestSeq method*), 561
- `__imul__()` (*rti.connexdds.ServiceRequestTimestampedSeq method*), 563
- `__imul__()` (*rti.connexdds.ServiceRequest.TopicSeq method*), 554
- `__imul__()` (*rti.connexdds.StringPairSeq method*), 573
- `__imul__()` (*rti.connexdds.StringSeq method*), 575
- `__imul__()` (*rti.connexdds.SubscriberSeq method*), 583
- `__imul__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 587
- `__imul__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 596
- `__imul__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 608
- `__imul__()` (*rti.connexdds.SubscriptionBuiltinTopicDataSeq method*), 623
- `__imul__()` (*rti.connexdds.SubscriptionBuiltinTopicData-TimestampedSeq method*), 625
- `__imul__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq method*), 617
- `__imul__()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 642
- `__imul__()` (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq method*), 651
- `__imul__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method*), 663
- `__imul__()` (*rti.connexdds.TopicBuiltinTopicDataSeq method*), 676
- `__imul__()` (*rti.connexdds.TopicBuiltinTopicData-TimestampedSeq method*), 679
- `__imul__()` (*rti.connexdds.TopicBuiltinTopicData.TopicSeq method*), 671
- `__imul__()` (*rti.connexdds.TopicSeq method*), 691
- `__imul__()` (*rti.connexdds.TransportInfoSeq method*), 700
- `__imul__()` (*rti.connexdds.TransportMulticastSeq method*), 707
- `__imul__()` (*rti.connexdds.TransportMulticastSettingsSeq method*), 709
- `__imul__()` (*rti.connexdds.TransportUnicastSettingsSeq method*), 714
- `__imul__()` (*rti.connexdds.Uint8Seq method*), 735
- `__imul__()` (*rti.connexdds.Uint16Seq method*), 727
- `__imul__()` (*rti.connexdds.Uint32Seq method*), 729
- `__imul__()` (*rti.connexdds.Uint64Seq method*), 732
- `__imul__()` (*rti.connexdds.UnionMemberSeq method*), 739
- `__imul__()` (*rti.connexdds.WcharSeq method*), 750
- `__imul__()` (*rti.connexdds.WstringSeq method*), 757
- `__index__()` (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind method*), 36
- `__index__()` (*rti.connexdds.CdrPaddingKind.CdrPaddingKind method*), 61
- `__index__()` (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 92
- `__index__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 136
- `__index__()` (*rti.connexdds.DestinationOrderKind.DestinationOrderKind method*), 143
- `__index__()` (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 145
- `__index__()` (*rti.connexdds.DurabilityKind.DurabilityKind method*), 180
- `__index__()` (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 246
- `__index__()` (*rti.connexdds.ExtensibilityKind.ExtensibilityKind method*), 269
- `__index__()` (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 283
- `__index__()` (*rti.connexdds.HistoryKind.HistoryKind method*), 289
- `__index__()` (*rti.connexdds.IgnoredEntityReplacement*

- tKind.IgnoredEntityReplacementKind* method), 306
- `__index__` () (*rti.connexdds.InstanceStateConsistencyKind* method), 314
- `__index__` () (*rti.connexdds.LivelinessKind.LivelinessKind* method), 329
- `__index__` () (*rti.connexdds.LocatorKind.LocatorKind* method), 336
- `__index__` () (*rti.connexdds.LogCategory.LogCategory* method), 342
- `__index__` () (*rti.connexdds.OwnershipKind.OwnershipKind* method), 368
- `__index__` () (*rti.connexdds.PersistentJournalKind* method), 412
- `__index__` () (*rti.connexdds.PersistentSynchronizationKind* method), 414
- `__index__` () (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind* method), 416
- `__index__` () (*rti.connexdds.PrintFormatKind.PrintFormatKind* method), 421
- `__index__` () (*rti.connexdds.PrintFormat.PrintFormat* method), 419
- `__index__` () (*rti.connexdds.PublishModeKind.PublishModeKind* method), 471
- `__index__` () (*rti.connexdds.ReliabilityKind.ReliabilityKind* method), 491
- `__index__` () (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* method), 494
- `__index__` () (*rti.connexdds.ServiceKind.ServiceKind* method), 522
- `__index__` () (*rti.connexdds.ServiceRequestId.ServiceRequestId* method), 559
- `__index__` () (*rti.connexdds.SyslogVerbosity* method), 629
- `__index__` () (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* method), 631
- `__index__` () (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind* method), 689
- `__index__` () (*rti.connexdds.TransportClassId.TransportClassId* method), 697
- `__index__` () (*rti.connexdds.TransportMulticastKind.TransportMulticastKind* method), 703
- `__index__` () (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind* method), 718
- `__index__` () (*rti.connexdds.TypeKind.TypeKind* method), 723
- `__index__` () (*rti.connexdds.Verbosity.Verbosity* method), 743
- `__index__` () (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* method), 753
- `__init__` () (*rti.connexdds.AcknowledgmentInfo* method), 35
- `__init__` () (*rti.connexdds.AcknowledgmentKind* method), 37
- `__init__` () (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind* method), 36
- `__init__` () (*rti.connexdds.AckResponseData* method), 34
- `__init__` () (*rti.connexdds.ACTEnumMember* method), 31
- `__init__` () (*rti.connexdds.ActivityContextMask* method), 38
- `__init__` () (*rti.connexdds.ACTMember* method), 32
- `__init__` () (*rti.connexdds.ACTUnionMember* method), 33
- `__init__` () (*rti.connexdds.AliasType* method), 41
- `__init__` () (*rti.connexdds.AllocationSettings* method), 41
- `__init__` () (*rti.connexdds.AnyDataReader* method), 42
- `__init__` () (*rti.connexdds.AnyDataReaderListener* method), 42
- `__init__` () (*rti.connexdds.AnyDataReaderSeq* method), 43
- `__init__` () (*rti.connexdds.AnyDataWriter* method), 45
- `__init__` () (*rti.connexdds.AnyDataWriterListener* method), 45
- `__init__` () (*rti.connexdds.AnyDataWriterSeq* method), 47
- `__init__` () (*rti.connexdds.AnyTopic* method), 48
- `__init__` () (*rti.connexdds.AnyTopicListener* method), 48
- `__init__` () (*rti.connexdds.AnyTopicSeq* method), 49
- `__init__` () (*rti.connexdds.ArrayType* method), 51
- `__init__` () (*rti.connexdds.AsynchronousPublisher* method), 51
- `__init__` () (*rti.connexdds.Availability* method), 52
- `__init__` () (*rti.connexdds.Batch* method), 53
- `__init__` () (*rti.connexdds.BoolSeq* method), 54
- `__init__` () (*rti.connexdds.BoolType* method), 56
- `__init__` () (*rti.connexdds.BuiltinProfiles* method), 56
- `__init__` () (*rti.connexdds.BuiltinTopicKey* method), 58
- `__init__` () (*rti.connexdds.BuiltinTopicReaderResourceLimits* method), 59
- `__init__` () (*rti.connexdds.ByteVector* method), 60
- `__init__` () (*rti.connexdds.CdrPaddingKind* method), 62
- `__init__` () (*rti.connexdds.CdrPaddingKind.CdrPaddingKind* method), 61
- `__init__` () (*rti.connexdds.ChannelSettings* method), 63
- `__init__` () (*rti.connexdds.ChannelSettingsSeq* method), 64
- `__init__` () (*rti.connexdds.CharSeq* method), 66
- `__init__` () (*rti.connexdds.CoherentAccess* method), 68
- `__init__` () (*rti.connexdds.CoherentSet* method), 68
- `__init__` () (*rti.connexdds.CoherentSetInfo* method), 69
- `__init__` () (*rti.connexdds.CollectionType* method), 69
- `__init__` () (*rti.connexdds.CompressionIdMask* method), 70
- `__init__` () (*rti.connexdds.CompressionSettings* method), 73
- `__init__` () (*rti.connexdds.Condition* method), 73
- `__init__` () (*rti.connexdds.ConditionSeq* method), 74
- `__init__` () (*rti.connexdds.ContentFilterBase* method), 75
- `__init__` () (*rti.connexdds.ContentFilteredTopic* method), 76
- `__init__` () (*rti.connexdds.ContentFilteredTopicSeq* method), 78
- `__init__` () (*rti.connexdds.ContentFilterProperty* method), 76
- `__init__` () (*rti.connexdds.Cookie* method), 80
- `__init__` () (*rti.connexdds.CookieSeq* method), 81
- `__init__` () (*rti.connexdds.CookieVector* method), 82
- `__init__` () (*rti.connexdds.Database* method), 140

`__init__()` (*rti.connexdds.DataReader* method), 85
`__init__()` (*rti.connexdds.DataReaderCacheStatus* method), 90
`__init__()` (*rti.connexdds.DataReaderInstanceRemovalKind* method), 93
`__init__()` (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind* method), 92
`__init__()` (*rti.connexdds.DataReaderListener* method), 94
`__init__()` (*rti.connexdds.DataReader.LoanedSample* method), 83
`__init__()` (*rti.connexdds.DataReader.LoanedSamples* method), 84
`__init__()` (*rti.connexdds.DataReaderProtocol* method), 95
`__init__()` (*rti.connexdds.DataReaderProtocolStatus* method), 95
`__init__()` (*rti.connexdds.DataReaderQos* method), 97
`__init__()` (*rti.connexdds.DataReaderResourceLimits* method), 104
`__init__()` (*rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings* method), 106
`__init__()` (*rti.connexdds.DataReader.Selector* method), 84
`__init__()` (*rti.connexdds.DataReaderSeq* method), 108
`__init__()` (*rti.connexdds.DataRepresentation* method), 109
`__init__()` (*rti.connexdds.DataState* method), 110
`__init__()` (*rti.connexdds.DataStateEx* method), 111
`__init__()` (*rti.connexdds.DataTag* method), 113
`__init__()` (*rti.connexdds.DataWriter* method), 114
`__init__()` (*rti.connexdds.DataWriterCacheStatus* method), 120
`__init__()` (*rti.connexdds.DataWriterListener* method), 121
`__init__()` (*rti.connexdds.DataWriterProtocol* method), 122
`__init__()` (*rti.connexdds.DataWriterProtocolStatus* method), 123
`__init__()` (*rti.connexdds.DataWriterQos* method), 125
`__init__()` (*rti.connexdds.DataWriterResourceLimits* method), 132
`__init__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind* method), 137
`__init__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind* method), 136
`__init__()` (*rti.connexdds.DataWriterSeq* method), 138
`__init__()` (*rti.connexdds.DataWriterShmemRefTransferModeSettings* method), 140
`__init__()` (*rti.connexdds.DataWriterTransferMode* method), 140
`__init__()` (*rti.connexdds.Deadline* method), 141
`__init__()` (*rti.connexdds.DestinationOrder* method), 142
`__init__()` (*rti.connexdds.DestinationOrderKind* method), 144
`__init__()` (*rti.connexdds.DestinationOrderKind.DestinationOrderKind* method), 143
`__init__()` (*rti.connexdds.DestinationOrderScopeKind* method), 146
`__init__()` (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind* method), 145
`__init__()` (*rti.connexdds.Discovery* method), 147
`__init__()` (*rti.connexdds.DiscoveryConfig* method), 147
`__init__()` (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 151
`__init__()` (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 154
`__init__()` (*rti.connexdds.DomainParticipant* method), 157
`__init__()` (*rti.connexdds.DomainParticipantConfigParams* method), 163
`__init__()` (*rti.connexdds.DomainParticipantFactoryQos* method), 164
`__init__()` (*rti.connexdds.DomainParticipantListener* method), 166
`__init__()` (*rti.connexdds.DomainParticipantProtocolStatus* method), 166
`__init__()` (*rti.connexdds.DomainParticipantQos* method), 166
`__init__()` (*rti.connexdds.DomainParticipantResourceLimits* method), 171
`__init__()` (*rti.connexdds.DomainParticipantSeq* method), 176
`__init__()` (*rti.connexdds.Durability* method), 178
`__init__()` (*rti.connexdds.DurabilityKind* method), 181
`__init__()` (*rti.connexdds.DurabilityKind.DurabilityKind* method), 180
`__init__()` (*rti.connexdds.DurabilityService* method), 181
`__init__()` (*rti.connexdds.Duration* method), 182
`__init__()` (*rti.connexdds.DynamicData* method), 221
`__init__()` (*rti.connexdds.DynamicData.ContentFilter* method), 184
`__init__()` (*rti.connexdds.DynamicData.ContentFilteredTopic* method), 185
`__init__()` (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 187
`__init__()` (*rti.connexdds.DynamicData.DataReader* method), 189
`__init__()` (*rti.connexdds.DynamicData.DataReaderListener* method), 194
`__init__()` (*rti.connexdds.DynamicData.DataReader.Selector* method), 188
`__init__()` (*rti.connexdds.DynamicData.DataReaderSeq* method), 196
`__init__()` (*rti.connexdds.DynamicData.DataWriter* method), 197
`__init__()` (*rti.connexdds.DynamicData.DataWriterListener* method), 205
`__init__()` (*rti.connexdds.DynamicData.DataWriterSeq* method), 207

__init__() (rti.connexdds.DynamicDataEncapsulationKind method), 247
 __init__() (rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method), 246
 __init__() (rti.connexdds.DynamicData.FieldsIterator method), 208
 __init__() (rti.connexdds.DynamicData.FieldsView method), 208
 __init__() (rti.connexdds.DynamicData.IndexIterator method), 209
 __init__() (rti.connexdds.DynamicDataInfo method), 248
 __init__() (rti.connexdds.DynamicData.ItemsIterator method), 209
 __init__() (rti.connexdds.DynamicData.ItemsView method), 210
 __init__() (rti.connexdds.DynamicData.ITopicDescription method), 209
 __init__() (rti.connexdds.DynamicData.LoanedSample method), 210
 __init__() (rti.connexdds.DynamicData.LoanedSamples method), 211
 __init__() (rti.connexdds.DynamicData.MemberInfo method), 248
 __init__() (rti.connexdds.DynamicData.NoOpDataReaderListener method), 211
 __init__() (rti.connexdds.DynamicData.NoOpDataWriterListener method), 212
 __init__() (rti.connexdds.DynamicData.NoOpTopicListener method), 213
 __init__() (rti.connexdds.DynamicData.Property method), 249
 __init__() (rti.connexdds.DynamicData.Sample method), 213
 __init__() (rti.connexdds.DynamicDataSeq method), 250
 __init__() (rti.connexdds.DynamicData.SharedSamples method), 214
 __init__() (rti.connexdds.DynamicData.TimestampedSeq method), 252
 __init__() (rti.connexdds.DynamicData.Topic method), 214
 __init__() (rti.connexdds.DynamicData.TopicDescription method), 216
 __init__() (rti.connexdds.DynamicData.TopicListener method), 216
 __init__() (rti.connexdds.DynamicData.TopicSeq method), 217
 __init__() (rti.connexdds.DynamicData.TopicTypeSupport method), 218
 __init__() (rti.connexdds.DynamicDataTypeSerializationProperty method), 254
 __init__() (rti.connexdds.DynamicData.ValidLoanedSamples method), 219
 __init__() (rti.connexdds.DynamicData.WriterContentFilter method), 219
 __init__() (rti.connexdds.DynamicData.WriterContentFilterHelper method), 220
 __init__() (rti.connexdds.DynamicType method), 255
 __init__() (rti.connexdds.DynamicTypePrintFormatProperty method), 256
 __init__() (rti.connexdds.EndpointGroup method), 256
 __init__() (rti.connexdds.EndpointGroupSeq method), 257
 __init__() (rti.connexdds.EndpointGroupVector method), 259
 __init__() (rti.connexdds.Entity method), 260
 __init__() (rti.connexdds.EntityFactory method), 260
 __init__() (rti.connexdds.EntityName method), 260
 __init__() (rti.connexdds.EntitySeq method), 261
 __init__() (rti.connexdds.EnumMember method), 263
 __init__() (rti.connexdds.EnumMemberSeq method), 264
 __init__() (rti.connexdds.EnumType method), 266
 __init__() (rti.connexdds.Event method), 267
 __init__() (rti.connexdds.EventCount32 method), 267
 __init__() (rti.connexdds.EventCount64 method), 267
 __init__() (rti.connexdds.ExclusiveArea method), 268
 __init__() (rti.connexdds.ExpressionProperty method), 268
 __init__() (rti.connexdds.ExtensibilityKind method), 270
 __init__() (rti.connexdds.ExtensibilityKind.ExtensibilityKind method), 269
 __init__() (rti.connexdds.Filter method), 271
 __init__() (rti.connexdds.FilterSampleInfo method), 272
 __init__() (rti.connexdds.Float32Seq method), 275
 __init__() (rti.connexdds.Float32Type method), 277
 __init__() (rti.connexdds.Float64Seq method), 278
 __init__() (rti.connexdds.Float64Type method), 279
 __init__() (rti.connexdds.Float128Seq method), 273
 __init__() (rti.connexdds.Float128Type method), 274
 __init__() (rti.connexdds.FlowController method), 280
 __init__() (rti.connexdds.FlowControllerProperty method), 280
 __init__() (rti.connexdds.FlowControllerSchedulingPolicy method), 284
 __init__() (rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method), 283
 __init__() (rti.connexdds.FlowControllerTokenBucketProperty method), 284
 __init__() (rti.connexdds.GenerationCount method), 285
 __init__() (rti.connexdds.GroupData method), 286
 __init__() (rti.connexdds.GuardCondition method), 286
 __init__() (rti.connexdds.Guid method), 287
 __init__() (rti.connexdds.History method), 288
 __init__() (rti.connexdds.HistoryKind method), 290
 __init__() (rti.connexdds.HistoryKind.HistoryKind method), 289
 __init__() (rti.connexdds.IAnyDataReader method), 290
 __init__() (rti.connexdds.IAnyDataReaderSeq method), 292
 __init__() (rti.connexdds.IAnyDataWriter method), 293
 __init__() (rti.connexdds.IAnyDataWriterSeq method), 295
 __init__() (rti.connexdds.IAnyTopic method), 296
 __init__() (rti.connexdds.IAnyTopicSeq method), 298
 __init__() (rti.connexdds.ICondition method), 299

__init__() (*rti.connexdds.IConditionSeq method*), 300
 __init__() (*rti.connexdds.IDataReader method*), 301
 __init__() (*rti.connexdds.IEntity method*), 302
 __init__() (*rti.connexdds.IEntitySeq method*), 303
 __init__() (*rti.connexdds.IgnoredEntityReplacementKind method*), 307
 __init__() (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 306
 __init__() (*rti.connexdds.InconsistentTopicStatus method*), 308
 __init__() (*rti.connexdds.InstanceHandle method*), 308
 __init__() (*rti.connexdds.InstanceHandleSeq method*), 309
 __init__() (*rti.connexdds.InstanceState method*), 311
 __init__() (*rti.connexdds.InstanceStateConsistencyKind method*), 314
 __init__() (*rti.connexdds.Int8Seq method*), 324
 __init__() (*rti.connexdds.Int8Type method*), 325
 __init__() (*rti.connexdds.Int16Seq method*), 315
 __init__() (*rti.connexdds.Int16Type method*), 317
 __init__() (*rti.connexdds.Int32Seq method*), 317
 __init__() (*rti.connexdds.Int32Type method*), 319
 __init__() (*rti.connexdds.Int32Vector method*), 320
 __init__() (*rti.connexdds.Int64Seq method*), 321
 __init__() (*rti.connexdds.Int64Type method*), 323
 __init__() (*rti.connexdds.InvalidLocalIdentityAdvanceNoticeStatus method*), 326
 __init__() (*rti.connexdds.IReadCondition method*), 305
 __init__() (*rti.connexdds.ITopicDescription method*), 305
 __init__() (*rti.connexdds.LatencyBudget method*), 326
 __init__() (*rti.connexdds.Lifespan method*), 326
 __init__() (*rti.connexdds.Liveliness method*), 327
 __init__() (*rti.connexdds.LivelinessChangedStatus method*), 328
 __init__() (*rti.connexdds.LivelinessKind method*), 330
 __init__() (*rti.connexdds.LivelinessKind.LivelinessKind method*), 329
 __init__() (*rti.connexdds.LivelinessLostStatus method*), 330
 __init__() (*rti.connexdds.LoanedDynamicData method*), 331
 __init__() (*rti.connexdds.Locator method*), 331
 __init__() (*rti.connexdds.LocatorFilter method*), 332
 __init__() (*rti.connexdds.LocatorFilterElement method*), 332
 __init__() (*rti.connexdds.LocatorFilterElementSeq method*), 334
 __init__() (*rti.connexdds.LocatorKind method*), 338
 __init__() (*rti.connexdds.LocatorKind.LocatorKind method*), 336
 __init__() (*rti.connexdds.LocatorSeq method*), 339
 __init__() (*rti.connexdds.LocatorVector method*), 340
 __init__() (*rti.connexdds.LogCategory method*), 343
 __init__() (*rti.connexdds.LogCategory.LogCategory method*), 342
 __init__() (*rti.connexdds.Logger method*), 344
 __init__() (*rti.connexdds.LongDouble method*), 345
 __init__() (*rti.connexdds.Member method*), 346
 __init__() (*rti.connexdds.MemberSeq method*), 348
 __init__() (*rti.connexdds.Monitoring method*), 349
 __init__() (*rti.connexdds.MonitoringDedicatedParticipantSettings method*), 350
 __init__() (*rti.connexdds.MonitoringDistributionSettings method*), 350
 __init__() (*rti.connexdds.MonitoringEventDistributionSettings method*), 351
 __init__() (*rti.connexdds.MonitoringLoggingDistributionSettings method*), 352
 __init__() (*rti.connexdds.MonitoringLoggingForwardingSettings method*), 352
 __init__() (*rti.connexdds.MonitoringMetricSelection method*), 353
 __init__() (*rti.connexdds.MonitoringMetricSelectionSeq method*), 354
 __init__() (*rti.connexdds.MonitoringPeriodicDistributionSettings method*), 356
 __init__() (*rti.connexdds.MonitoringTelemetryData method*), 356
 __init__() (*rti.connexdds.MulticastMapping method*), 357
 __init__() (*rti.connexdds.MulticastMappingSeq method*), 359
 __init__() (*rti.connexdds.MultiChannel method*), 357
 __init__() (*rti.connexdds.NoOpAnyDataReaderListener method*), 360
 __init__() (*rti.connexdds.NoOpAnyDataWriterListener method*), 361
 __init__() (*rti.connexdds.NoOpAnyTopicListener method*), 362
 __init__() (*rti.connexdds.NoOpDataReaderListener method*), 362
 __init__() (*rti.connexdds.NoOpDataWriterListener method*), 363
 __init__() (*rti.connexdds.NoOpDomainParticipantListener method*), 364
 __init__() (*rti.connexdds.NoOpPublisherListener method*), 365
 __init__() (*rti.connexdds.NoOpSubscriberListener method*), 365
 __init__() (*rti.connexdds.NoOpTopicListener method*), 365
 __init__() (*rti.connexdds.OfferedDeadlineMissedStatus method*), 366
 __init__() (*rti.connexdds.OfferedIncompatibleQosStatus method*), 366
 __init__() (*rti.connexdds.Ownership method*), 367
 __init__() (*rti.connexdds.OwnershipKind method*), 368
 __init__() (*rti.connexdds.OwnershipKind.OwnershipKind method*), 368
 __init__() (*rti.connexdds.OwnershipStrength method*), 369
 __init__() (*rti.connexdds.ParticipantBuiltinTopicData method*), 405
 __init__() (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilter method*), 370

<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic</i> method), 370	<code>icDescription</code> method), 401
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq</i> method), 372	<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.TopicListener</i> method), 401
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.DataReader</i> method), 375	<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.TopicSeq</i> method), 402
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener</i> method), 380	<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.ValidLoanedSamples</i> method), 404
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector</i> method), 373	<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter</i> method), 404
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq</i> method), 381	<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilterHelper</i> method), 405
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.DataWriter</i> method), 383	<code>__init__</code> () (<i>rti.connexdds.Partition</i> method), 411
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.DataWriterListener</i> method), 391	<code>__init__</code> () (<i>rti.connexdds.PersistentJournalKind</i> method), 412
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq</i> method), 393	<code>__init__</code> () (<i>rti.connexdds.PersistentStorageSettings</i> method), 413
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.ITopicDescription</i> method), 395	<code>__init__</code> () (<i>rti.connexdds.PersistentSynchronizationKind</i> method), 414
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.LoanedSample</i> method), 395	<code>__init__</code> () (<i>rti.connexdds.Presentation</i> method), 415
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples</i> method), 395	<code>__init__</code> () (<i>rti.connexdds.PresentationAccessScopeKind</i> method), 417
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener</i> method), 396	<code>__init__</code> () (<i>rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind</i> method), 416
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.NoOpDataWriterListener</i> method), 397	<code>__init__</code> () (<i>rti.connexdds.PrintFormat</i> method), 420
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.NoOpTopicListener</i> method), 398	<code>__init__</code> () (<i>rti.connexdds.PrintFormatKind</i> method), 422
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.Sample</i> method), 399	<code>__init__</code> () (<i>rti.connexdds.PrintFormatKind.PrintFormatKind</i> method), 421
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicDataSeq</i> method), 407	<code>__init__</code> () (<i>rti.connexdds.PrintFormat.PrintFormat</i> method), 419
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.SharedSamples</i> method), 399	<code>__init__</code> () (<i>rti.connexdds.PrintFormatProperty</i> method), 422
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq</i> method), 409	<code>__init__</code> () (<i>rti.connexdds.ProductVersion</i> method), 423
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.Topic</i> method), 400	<code>__init__</code> () (<i>rti.connexdds.Property</i> method), 424
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.TopicDescription</i> method), 401	<code>__init__</code> () (<i>rti.connexdds.ProtocolVersion</i> method), 425
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.TopicListener</i> method), 401	<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData</i> method), 462
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.TopicSeq</i> method), 402	<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilter</i> method), 426
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.ValidLoanedSamples</i> method), 404	<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic</i> method), 426
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter</i> method), 404	<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq</i> method), 428
<code>__init__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilterHelper</i> method), 405	<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.DataReader</i> method), 431
<code>__init__</code> () (<i>rti.connexdds.Partition</i> method), 411	<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.DataReaderListener</i> method), 436
<code>__init__</code> () (<i>rti.connexdds.PersistentJournalKind</i> method), 412	
<code>__init__</code> () (<i>rti.connexdds.PersistentStorageSettings</i> method), 413	
<code>__init__</code> () (<i>rti.connexdds.PersistentSynchronizationKind</i> method), 414	
<code>__init__</code> () (<i>rti.connexdds.Presentation</i> method), 415	
<code>__init__</code> () (<i>rti.connexdds.PresentationAccessScopeKind</i> method), 417	
<code>__init__</code> () (<i>rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind</i> method), 416	
<code>__init__</code> () (<i>rti.connexdds.PrintFormat</i> method), 420	
<code>__init__</code> () (<i>rti.connexdds.PrintFormatKind</i> method), 422	
<code>__init__</code> () (<i>rti.connexdds.PrintFormatKind.PrintFormatKind</i> method), 421	
<code>__init__</code> () (<i>rti.connexdds.PrintFormat.PrintFormat</i> method), 419	
<code>__init__</code> () (<i>rti.connexdds.PrintFormatProperty</i> method), 422	
<code>__init__</code> () (<i>rti.connexdds.ProductVersion</i> method), 423	
<code>__init__</code> () (<i>rti.connexdds.Property</i> method), 424	
<code>__init__</code> () (<i>rti.connexdds.ProtocolVersion</i> method), 425	
<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData</i> method), 462	
<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilter</i> method), 426	
<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic</i> method), 426	
<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq</i> method), 428	
<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.DataReader</i> method), 431	
<code>__init__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.DataReaderListener</i> method), 436	

- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector* method), 430
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 438
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter* method), 440
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterListener* method), 447
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 450
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.ITopicDescription* method), 451
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.LoanedSample* method), 451
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.LoanedSamples* method), 452
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener* method), 452
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener* method), 453
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.NoOpTopicListener* method), 455
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.Sample* method), 455
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 465
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.SharedSamples* method), 456
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.TimestampedSeq* method), 467
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.Topic* method), 456
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicDescription* method), 457
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicListener* method), 457
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 459
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.ValidLoanedSamples* method), 460
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.WriterContentFilter* method), 460
- `__init__()` (*rti.connexdds.PublicationBuiltinTopicData.WriterContentFilterHelper* method), 461
- `__init__()` (*rti.connexdds.PublicationMatchedStatus* method), 469
- `__init__()` (*rti.connexdds.Publisher* method), 472
- `__init__()` (*rti.connexdds.PublisherListener* method), 473
- `__init__()` (*rti.connexdds.PublisherQos* method), 474
- `__init__()` (*rti.connexdds.PublisherSeq* method), 477
- `__init__()` (*rti.connexdds.PublishMode* method), 469
- `__init__()` (*rti.connexdds.PublishModeKind* method), 471
- `__init__()` (*rti.connexdds.PublishModeKind.PublishModeKind* method), 471
- `__init__()` (*rti.connexdds.QosPolicyCount* method), 478
- `__init__()` (*rti.connexdds.QosPolicyCountSeq* method), 479
- `__init__()` (*rti.connexdds.QosPrintFormat* method), 481
- `__init__()` (*rti.connexdds.QosProvider* method), 481
- `__init__()` (*rti.connexdds.QosProviderParams* method), 484
- `__init__()` (*rti.connexdds.Query* method), 485
- `__init__()` (*rti.connexdds.QueryCondition* method), 486
- `__init__()` (*rti.connexdds.Rank* method), 486
- `__init__()` (*rti.connexdds.ReadCondition* method), 487
- `__init__()` (*rti.connexdds.ReaderDataLifecycle* method), 488
- `__init__()` (*rti.connexdds.ReceiverPool* method), 489
- `__init__()` (*rti.connexdds.Reliability* method), 489
- `__init__()` (*rti.connexdds.ReliabilityKind* method), 491
- `__init__()` (*rti.connexdds.ReliabilityKind.ReliabilityKind* method), 491
- `__init__()` (*rti.connexdds.ReliableReaderActivityChangedStatus* method), 492
- `__init__()` (*rti.connexdds.ReliableWriterCacheChangedStatus* method), 492
- `__init__()` (*rti.connexdds.RemoteParticipantPurgeKind* method), 495
- `__init__()` (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* method), 494
- `__init__()` (*rti.connexdds.RequestedDeadlineMissedStatus* method), 495
- `__init__()` (*rti.connexdds.RequestedIncompatibleQosStatus* method), 496
- `__init__()` (*rti.connexdds.ResourceLimits* method), 496
- `__init__()` (*rti.connexdds.RtpsReliableReaderProtocol* method), 497
- `__init__()` (*rti.connexdds.RtpsReliableWriterProtocol* method), 498
- `__init__()` (*rti.connexdds.RtpsReservedPortKindMask* method), 501
- `__init__()` (*rti.connexdds.RtpsWellKnownPorts* method), 504
- `__init__()` (*rti.connexdds.SampleFlag* method), 505
- `__init__()` (*rti.connexdds.SampleIdentity* method), 507
- `__init__()` (*rti.connexdds.SampleInfo* method), 508

[__init__ \(\) \(rti.connexdds.SampleLostState method\), 511](#)
[__init__ \(\) \(rti.connexdds.SampleLostStatus method\), 513](#)
[__init__ \(\) \(rti.connexdds.SampleRejectedState method\), 514](#)
[__init__ \(\) \(rti.connexdds.SampleRejectedStatus method\), 516](#)
[__init__ \(\) \(rti.connexdds.SampleState method\), 517](#)
[__init__ \(\) \(rti.connexdds.SequenceNumber method\), 519](#)
[__init__ \(\) \(rti.connexdds.SequenceType method\), 520](#)
[__init__ \(\) \(rti.connexdds.Service method\), 521](#)
[__init__ \(\) \(rti.connexdds.ServiceKind method\), 523](#)
[__init__ \(\) \(rti.connexdds.ServiceKind.ServiceKind method\), 522](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest method\), 557](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.AcceptedStatus method\), 558](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.ContentFilter method\), 523](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.ContentFilteredTopic method\), 524](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.ContentFiltered-TopicSeq method\), 526](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.DataReader method\), 529](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.DataReaderListener method\), 533](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.DataReader.Selector method\), 527](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.DataReaderSeq method\), 535](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.DataWriter method\), 536](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.DataWriterListener method\), 544](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.DataWriterSeq method\), 546](#)
[__init__ \(\) \(rti.connexdds.ServiceRequestId method\), 560](#)
[__init__ \(\) \(rti.connexdds.ServiceRequestId.ServiceRequestId method\), 559](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.ITopicDescription method\), 547](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.LoanedSample method\), 547](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.LoanedSamples method\), 548](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.NoOpDataReaderListener method\), 548](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.NoOpDataWriterListener method\), 549](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.NoOpTopicListener method\), 551](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.Sample method\), 551](#)
[__init__ \(\) \(rti.connexdds.ServiceRequestSeq method\), 561](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.SharedSamples method\), 552](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.TimestampedSeq method\), 563](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.Topic method\), 552](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.TopicDescription method\), 553](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.TopicListener method\), 553](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.TopicSeq method\), 554](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.ValidLoanedSamples method\), 556](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.WriterContentFilter method\), 556](#)
[__init__ \(\) \(rti.connexdds.ServiceRequest.WriterContentFilterHelper method\), 557](#)
[__init__ \(\) \(rti.connexdds.StatusCondition method\), 565](#)
[__init__ \(\) \(rti.connexdds.StatusMask method\), 567](#)
[__init__ \(\) \(rti.connexdds.StreamKind method\), 570](#)
[__init__ \(\) \(rti.connexdds.StringMap method\), 572](#)
[__init__ \(\) \(rti.connexdds.StringPairSeq method\), 573](#)
[__init__ \(\) \(rti.connexdds.StringSeq method\), 575](#)
[__init__ \(\) \(rti.connexdds.StringType method\), 577](#)
[__init__ \(\) \(rti.connexdds.StructType method\), 577](#)
[__init__ \(\) \(rti.connexdds.Subscriber method\), 578](#)
[__init__ \(\) \(rti.connexdds.SubscriberListener method\), 580](#)
[__init__ \(\) \(rti.connexdds.SubscriberQos method\), 580](#)
[__init__ \(\) \(rti.connexdds.SubscriberSeq method\), 583](#)
[__init__ \(\) \(rti.connexdds.SubscriptionBuiltinTopicData method\), 620](#)
[__init__ \(\) \(rti.connexdds.SubscriptionBuiltinTopicData.ContentFilter method\), 584](#)
[__init__ \(\) \(rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic method\), 585](#)
[__init__ \(\) \(rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method\), 587](#)
[__init__ \(\) \(rti.connexdds.SubscriptionBuiltinTopicData.DataReader method\), 590](#)
[__init__ \(\) \(rti.connexdds.SubscriptionBuiltinTopicData.DataReaderListener method\), 594](#)
[__init__ \(\) \(rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector method\), 588](#)
[__init__ \(\) \(rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq method\), 596](#)

`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* method), 598
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener* method), 606
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 608
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ITopicDescription* method), 609
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.LoanedSample* method), 610
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples* method), 610
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener* method), 611
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener* method), 612
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpTopicListener* method), 613
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.Sample* method), 614
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 623
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.SharedSamples* method), 614
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TimestampedSeq* method), 626
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.Topic* method), 615
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription* method), 616
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicListener* method), 616
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 617
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ValidLoanedSamples* method), 619
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter* method), 619
`__init__()` (*rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper* method), 620
`__init__()` (*rti.connexdds.SubscriptionMatchedStatus* method), 627
`__init__()` (*rti.connexdds.SuspendedPublication* method), 628
`__init__()` (*rti.connexdds.SyslogVerbosity* method), 629
`__init__()` (*rti.connexdds.SystemResourceLimits* method), 630
`__init__()` (*rti.connexdds.ThreadContext* method), 630
`__init__()` (*rti.connexdds.ThreadSettings* method), 630
`__init__()` (*rti.connexdds.ThreadSettingsCpuRotationKind* method), 632
`__init__()` (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* method), 631
`__init__()` (*rti.connexdds.ThreadSettingsKindMask* method), 634
`__init__()` (*rti.connexdds.Time* method), 636
`__init__()` (*rti.connexdds.TimeBasedFilter* method), 638
`__init__()` (*rti.connexdds.Topic* method), 638
`__init__()` (*rti.connexdds.TopicBuiltinTopicData* method), 674
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilter* method), 640
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic* method), 640
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 642
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.DataReader* method), 645
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.DataReaderListener* method), 650
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector* method), 643
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq* method), 651
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriter* method), 653
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterListener* method), 661
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq* method), 663
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.ITopicDescription* method), 664
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.LoanedSample* method), 664
`__init__()` (*rti.connexdds.TopicBuiltinTopicData.LoanedSamples*

- `method`), 665
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.NoOp-DataReaderListener method*), 665
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.NoOp-DataWriterListener method*), 666
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.NoOpTopicListener method*), 668
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.Sample method*), 668
- `__init__` () (*rti.connexdds.TopicBuiltinTopicDataSeq method*), 676
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.SharedSamples method*), 669
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.TimestampedSeq method*), 679
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.Topic method*), 669
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.TopicDescription method*), 670
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.TopicListener method*), 670
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq method*), 671
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.ValidLoanedSamples method*), 673
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilter method*), 673
- `__init__` () (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilterHelper method*), 674
- `__init__` () (*rti.connexdds.TopicData method*), 681
- `__init__` () (*rti.connexdds.TopicDescription method*), 681
- `__init__` () (*rti.connexdds.TopicListener method*), 682
- `__init__` () (*rti.connexdds.TopicQos method*), 682
- `__init__` () (*rti.connexdds.TopicQuery method*), 686
- `__init__` () (*rti.connexdds.TopicQueryData method*), 687
- `__init__` () (*rti.connexdds.TopicQueryDispatch method*), 687
- `__init__` () (*rti.connexdds.TopicQuerySelection method*), 688
- `__init__` () (*rti.connexdds.TopicQuerySelectionKind method*), 689
- `__init__` () (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 689
- `__init__` () (*rti.connexdds.TopicSeq method*), 691
- `__init__` () (*rti.connexdds.TransportBuiltin method*), 692
- `__init__` () (*rti.connexdds.TransportBuiltinMask method*), 693
- `__init__` () (*rti.connexdds.TransportClassId method*), 698
- `__init__` () (*rti.connexdds.TransportClassId.TransportClassId method*), 697
- `__init__` () (*rti.connexdds.TransportInfo method*), 699
- `__init__` () (*rti.connexdds.TransportInfoSeq method*), 700
- `__init__` () (*rti.connexdds.TransportInfoVector method*), 701
- `__init__` () (*rti.connexdds.TransportMulticast method*), 702
- `__init__` () (*rti.connexdds.TransportMulticastKind method*), 704
- `__init__` () (*rti.connexdds.TransportMulticastKind.TransportMulticastKind method*), 703
- `__init__` () (*rti.connexdds.TransportMulticastMapping method*), 705
- `__init__` () (*rti.connexdds.TransportMulticastMappingFunction method*), 705
- `__init__` () (*rti.connexdds.TransportMulticastSeq method*), 707
- `__init__` () (*rti.connexdds.TransportMulticastSettings method*), 708
- `__init__` () (*rti.connexdds.TransportMulticastSettingsSeq method*), 709
- `__init__` () (*rti.connexdds.TransportPriority method*), 711
- `__init__` () (*rti.connexdds.TransportSelection method*), 712
- `__init__` () (*rti.connexdds.TransportUnicast method*), 712
- `__init__` () (*rti.connexdds.TransportUnicastSettings method*), 713
- `__init__` () (*rti.connexdds.TransportUnicastSettingsSeq method*), 714
- `__init__` () (*rti.connexdds.TriggeredConditions method*), 716
- `__init__` () (*rti.connexdds.TriggeredConditionsIterator method*), 716
- `__init__` () (*rti.connexdds.TypeConsistencyEnforcement method*), 716
- `__init__` () (*rti.connexdds.TypeConsistencyEnforcementKind method*), 719
- `__init__` () (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 718
- `__init__` () (*rti.connexdds.TypeKind method*), 725
- `__init__` () (*rti.connexdds.TypeKind.TypeKind method*), 723
- `__init__` () (*rti.connexdds.TypeSupport method*), 726
- `__init__` () (*rti.connexdds.Uint8Seq method*), 735
- `__init__` () (*rti.connexdds.Uint8Type method*), 736
- `__init__` () (*rti.connexdds.Uint16Seq method*), 727
- `__init__` () (*rti.connexdds.Uint16Type method*), 729
- `__init__` () (*rti.connexdds.Uint32Seq method*), 730
- `__init__` () (*rti.connexdds.Uint32Type method*), 731
- `__init__` () (*rti.connexdds.Uint64Seq method*), 732
- `__init__` () (*rti.connexdds.Uint64Type method*), 734
- `__init__` () (*rti.connexdds.UnidimensionalCollectionType method*), 737
- `__init__` () (*rti.connexdds.UnionMember method*), 737
- `__init__` () (*rti.connexdds.UnionMemberSeq method*), 739
- `__init__` () (*rti.connexdds.UnionType method*), 740
- `__init__` () (*rti.connexdds.UserData method*), 741
- `__init__` () (*rti.connexdds.UserDataSample method*), 741
- `__init__` () (*rti.connexdds.VendorId method*), 742
- `__init__` () (*rti.connexdds.Verbosity method*), 744
- `__init__` () (*rti.connexdds.Verbosity.Verbosity method*), 743

- `__init__()` (*rti.connexdds.ViewState method*), 745
- `__init__()` (*rti.connexdds.WaitSet method*), 747
- `__init__()` (*rti.connexdds.WaitSetProperty method*), 749
- `__init__()` (*rti.connexdds.WcharSeq method*), 750
- `__init__()` (*rti.connexdds.WcharType method*), 751
- `__init__()` (*rti.connexdds.WireProtocol method*), 752
- `__init__()` (*rti.connexdds.WireProtocolAutoKind method*), 754
- `__init__()` (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 753
- `__init__()` (*rti.connexdds.WriteParams method*), 755
- `__init__()` (*rti.connexdds.WriterDataLifecycle method*), 756
- `__init__()` (*rti.connexdds.WstringSeq method*), 757
- `__init__()` (*rti.connexdds.WstringType method*), 747
- `__init__()` (*rti.rpc.Replier method*), 765
- `__init__()` (*rti.rpc.Requester method*), 761
- `__init__()` (*rti.rpc.SimpleReplier method*), 768
- `__int__()` (*rti.connexdds.AcknowledgmentKind method*), 37
- `__int__()` (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind method*), 36
- `__int__()` (*rti.connexdds.ActivityContextMask method*), 38
- `__int__()` (*rti.connexdds.CdrPaddingKind method*), 62
- `__int__()` (*rti.connexdds.CdrPaddingKind.CdrPaddingKind method*), 61
- `__int__()` (*rti.connexdds.CompressionIdMask method*), 71
- `__int__()` (*rti.connexdds.DataReaderInstanceRemovalKind method*), 93
- `__int__()` (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 92
- `__int__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind method*), 137
- `__int__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 136
- `__int__()` (*rti.connexdds.DestinationOrderKind method*), 144
- `__int__()` (*rti.connexdds.DestinationOrderKind.DestinationOrderKind method*), 143
- `__int__()` (*rti.connexdds.DestinationOrderScopeKind method*), 146
- `__int__()` (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 145
- `__int__()` (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method*), 151
- `__int__()` (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method*), 155
- `__int__()` (*rti.connexdds.DurabilityKind method*), 181
- `__int__()` (*rti.connexdds.DurabilityKind.DurabilityKind method*), 180
- `__int__()` (*rti.connexdds.Duration method*), 183
- `__int__()` (*rti.connexdds.DynamicDataEncapsulationKind method*), 247
- `__int__()` (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 246
- `__int__()` (*rti.connexdds.EnumMember method*), 263
- `__int__()` (*rti.connexdds.ExtensibilityKind method*), 270
- `__int__()` (*rti.connexdds.ExtensibilityKind.ExtensibilityKind method*), 269
- `__int__()` (*rti.connexdds.FlowControllerSchedulingPolicy method*), 284
- `__int__()` (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 283
- `__int__()` (*rti.connexdds.HistoryKind method*), 290
- `__int__()` (*rti.connexdds.HistoryKind.HistoryKind method*), 289
- `__int__()` (*rti.connexdds.IgnoredEntityReplacementKind method*), 307
- `__int__()` (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 306
- `__int__()` (*rti.connexdds.InstanceState method*), 312
- `__int__()` (*rti.connexdds.InstanceStateConsistencyKind method*), 314
- `__int__()` (*rti.connexdds.LivelinessKind method*), 330
- `__int__()` (*rti.connexdds.LivelinessKind.LivelinessKind method*), 329
- `__int__()` (*rti.connexdds.LocatorKind method*), 338
- `__int__()` (*rti.connexdds.LocatorKind.LocatorKind method*), 336
- `__int__()` (*rti.connexdds.LogCategory method*), 343
- `__int__()` (*rti.connexdds.LogCategory.LogCategory method*), 342
- `__int__()` (*rti.connexdds.OwnershipKind method*), 369
- `__int__()` (*rti.connexdds.OwnershipKind.OwnershipKind method*), 368
- `__int__()` (*rti.connexdds.PersistentJournalKind method*), 412
- `__int__()` (*rti.connexdds.PersistentSynchronizationKind method*), 414
- `__int__()` (*rti.connexdds.PresentationAccessScopeKind method*), 418
- `__int__()` (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 417
- `__int__()` (*rti.connexdds.PrintFormat method*), 420
- `__int__()` (*rti.connexdds.PrintFormatKind method*), 422
- `__int__()` (*rti.connexdds.PrintFormatKind.PrintFormatKind method*), 421
- `__int__()` (*rti.connexdds.PrintFormat.PrintFormat method*), 419
- `__int__()` (*rti.connexdds.PublishModeKind method*), 472
- `__int__()` (*rti.connexdds.PublishModeKind.PublishModeKind method*), 471
- `__int__()` (*rti.connexdds.ReliabilityKind method*), 492
- `__int__()` (*rti.connexdds.ReliabilityKind.ReliabilityKind method*), 491

`__int__()` (*rti.connexdds.RemoteParticipantPurgeKind* method), 495
`__int__()` (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* method), 494
`__int__()` (*rti.connexdds.RtpsReservedPortKindMask* method), 501
`__int__()` (*rti.connexdds.SampleFlag* method), 505
`__int__()` (*rti.connexdds.SampleLostState* method), 511
`__int__()` (*rti.connexdds.SampleRejectedState* method), 514
`__int__()` (*rti.connexdds.SampleState* method), 517
`__int__()` (*rti.connexdds.SequenceNumber* method), 519
`__int__()` (*rti.connexdds.ServiceKind* method), 523
`__int__()` (*rti.connexdds.ServiceKind.ServiceKind* method), 522
`__int__()` (*rti.connexdds.ServiceRequestId* method), 560
`__int__()` (*rti.connexdds.ServiceRequestId.ServiceRequestId* method), 559
`__int__()` (*rti.connexdds.StatusMask* method), 567
`__int__()` (*rti.connexdds.StreamKind* method), 570
`__int__()` (*rti.connexdds.SyslogVerbosity* method), 629
`__int__()` (*rti.connexdds.ThreadSettingsCpuRotationKind* method), 633
`__int__()` (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* method), 632
`__int__()` (*rti.connexdds.ThreadSettingsKindMask* method), 634
`__int__()` (*rti.connexdds.TopicQuerySelectionKind* method), 690
`__int__()` (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind* method), 689
`__int__()` (*rti.connexdds.TransportBuiltinMask* method), 694
`__int__()` (*rti.connexdds.TransportClassId* method), 698
`__int__()` (*rti.connexdds.TransportClassId.TransportClassId* method), 697
`__int__()` (*rti.connexdds.TransportMulticastKind* method), 704
`__int__()` (*rti.connexdds.TransportMulticastKind.TransportMulticastKind* method), 703
`__int__()` (*rti.connexdds.TypeConsistencyEnforcementKind* method), 719
`__int__()` (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind* method), 718
`__int__()` (*rti.connexdds.TypeKind* method), 725
`__int__()` (*rti.connexdds.TypeKind.TypeKind* method), 723
`__int__()` (*rti.connexdds.Verbosity* method), 744
`__int__()` (*rti.connexdds.Verbosity.Verbosity* method), 743
`__int__()` (*rti.connexdds.ViewState* method), 745
`__int__()` (*rti.connexdds.WireProtocolAutoKind* method), 754
`__int__()` (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* method), 753
`__ior__()` (*rti.connexdds.ActivityContextMask* method), 39
`__ior__()` (*rti.connexdds.CompressionIdMask* method), 71
`__ior__()` (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 151
`__ior__()` (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 155
`__ior__()` (*rti.connexdds.InstanceState* method), 312
`__ior__()` (*rti.connexdds.RtpsReservedPortKindMask* method), 501
`__ior__()` (*rti.connexdds.SampleFlag* method), 505
`__ior__()` (*rti.connexdds.SampleLostState* method), 511
`__ior__()` (*rti.connexdds.SampleRejectedState* method), 514
`__ior__()` (*rti.connexdds.SampleState* method), 517
`__ior__()` (*rti.connexdds.StatusMask* method), 568
`__ior__()` (*rti.connexdds.StreamKind* method), 570
`__ior__()` (*rti.connexdds.ThreadSettingsKindMask* method), 634
`__ior__()` (*rti.connexdds.TransportBuiltinMask* method), 694
`__ior__()` (*rti.connexdds.ViewState* method), 745
`__irshift__()` (*rti.connexdds.ActivityContextMask* method), 39
`__irshift__()` (*rti.connexdds.CompressionIdMask* method), 71
`__irshift__()` (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 152
`__irshift__()` (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 155
`__irshift__()` (*rti.connexdds.InstanceState* method), 312
`__irshift__()` (*rti.connexdds.RtpsReservedPortKindMask* method), 502
`__irshift__()` (*rti.connexdds.SampleFlag* method), 506
`__irshift__()` (*rti.connexdds.SampleLostState* method), 511
`__irshift__()` (*rti.connexdds.SampleRejectedState* method), 514
`__irshift__()` (*rti.connexdds.SampleState* method), 517
`__irshift__()` (*rti.connexdds.StatusMask* method), 568
`__irshift__()` (*rti.connexdds.StreamKind* method), 570
`__irshift__()` (*rti.connexdds.ThreadSettingsKindMask* method), 634
`__irshift__()` (*rti.connexdds.TransportBuiltinMask* method), 694
`__irshift__()` (*rti.connexdds.ViewState* method), 745
`__isub__()` (*rti.connexdds.Duration* method), 183
`__isub__()` (*rti.connexdds.SequenceNumber* method), 519
`__isub__()` (*rti.connexdds.Time* method), 637
`__isub__()` (*rti.connexdds.WaitSet* method), 747
`__iter__()` (*rti.connexdds.AnyDataReaderSeq* method), 44
`__iter__()` (*rti.connexdds.AnyDataWriterSeq* method), 47
`__iter__()` (*rti.connexdds.AnyTopicSeq* method), 49
`__iter__()` (*rti.connexdds.BoolSeq* method), 55
`__iter__()` (*rti.connexdds.ByteVector* method), 60
`__iter__()` (*rti.connexdds.ChannelSettingsSeq* method), 64
`__iter__()` (*rti.connexdds.CharSeq* method), 66
`__iter__()` (*rti.connexdds.ConditionSeq* method), 74
`__iter__()` (*rti.connexdds.ContentFilteredTopicSeq* method), 78

- `__iter__()` (*rti.connexdds.CookieSeq* method), 81
- `__iter__()` (*rti.connexdds.CookieVector* method), 82
- `__iter__()` (*rti.connexdds.DataReader.LoanedSample* method), 83
- `__iter__()` (*rti.connexdds.DataReader.LoanedSamples* method), 84
- `__iter__()` (*rti.connexdds.DataReaderSeq* method), 108
- `__iter__()` (*rti.connexdds.DataWriterSeq* method), 138
- `__iter__()` (*rti.connexdds.DomainParticipantSeq* method), 177
- `__iter__()` (*rti.connexdds.DynamicData* method), 221
- `__iter__()` (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 187
- `__iter__()` (*rti.connexdds.DynamicData.DataReaderSeq* method), 196
- `__iter__()` (*rti.connexdds.DynamicData.DataWriterSeq* method), 207
- `__iter__()` (*rti.connexdds.DynamicData.FieldsIterator* method), 208
- `__iter__()` (*rti.connexdds.DynamicData.FieldsView* method), 208
- `__iter__()` (*rti.connexdds.DynamicData.IndexIterator* method), 209
- `__iter__()` (*rti.connexdds.DynamicData.ItemsIterator* method), 209
- `__iter__()` (*rti.connexdds.DynamicData.ItemsView* method), 210
- `__iter__()` (*rti.connexdds.DynamicData.LoanedSample* method), 210
- `__iter__()` (*rti.connexdds.DynamicData.LoanedSamples* method), 211
- `__iter__()` (*rti.connexdds.DynamicData.Sample* method), 214
- `__iter__()` (*rti.connexdds.DynamicDataSeq* method), 250
- `__iter__()` (*rti.connexdds.DynamicData.SharedSamples* method), 214
- `__iter__()` (*rti.connexdds.DynamicData.TimestampedSeq* method), 252
- `__iter__()` (*rti.connexdds.DynamicData.TopicSeq* method), 217
- `__iter__()` (*rti.connexdds.DynamicData.ValidLoanedSamples* method), 219
- `__iter__()` (*rti.connexdds.EndpointGroupSeq* method), 257
- `__iter__()` (*rti.connexdds.EndpointGroupVector* method), 259
- `__iter__()` (*rti.connexdds.EntitySeq* method), 262
- `__iter__()` (*rti.connexdds.EnumMemberSeq* method), 264
- `__iter__()` (*rti.connexdds.Filter* method), 271
- `__iter__()` (*rti.connexdds.Float32Seq* method), 275
- `__iter__()` (*rti.connexdds.Float64Seq* method), 278
- `__iter__()` (*rti.connexdds.Float128Seq* method), 273
- `__iter__()` (*rti.connexdds.GroupData* method), 286
- `__iter__()` (*rti.connexdds.IAnyDataReaderSeq* method), 292
- `__iter__()` (*rti.connexdds.IAnyDataWriterSeq* method), 295
- `__iter__()` (*rti.connexdds.IAnyTopicSeq* method), 298
- `__iter__()` (*rti.connexdds.IConditionSeq* method), 300
- `__iter__()` (*rti.connexdds.IEntitySeq* method), 303
- `__iter__()` (*rti.connexdds.InstanceHandleSeq* method), 310
- `__iter__()` (*rti.connexdds.Int8Seq* method), 324
- `__iter__()` (*rti.connexdds.Int16Seq* method), 315
- `__iter__()` (*rti.connexdds.Int32Seq* method), 318
- `__iter__()` (*rti.connexdds.Int32Vector* method), 320
- `__iter__()` (*rti.connexdds.Int64Seq* method), 321
- `__iter__()` (*rti.connexdds.LocatorFilterElementSeq* method), 334
- `__iter__()` (*rti.connexdds.LocatorSeq* method), 339
- `__iter__()` (*rti.connexdds.LocatorVector* method), 341
- `__iter__()` (*rti.connexdds.MemberSeq* method), 348
- `__iter__()` (*rti.connexdds.MonitoringMetricSelectionSeq* method), 354
- `__iter__()` (*rti.connexdds.MulticastMappingSeq* method), 359
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 372
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 382
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 393
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicData.LoanedSample* method), 395
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples* method), 396
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicData.Sample* method), 399
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 407
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicData.SharedSamples* method), 399
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq* method), 410
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 402
- `__iter__()` (*rti.connexdds.ParticipantBuiltinTopicData.ValidLoanedSamples* method), 404
- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 428
- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 438
- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 450
- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicData.LoanedSample* method), 451

- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicData.LoanedSamples* method), 452
- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicData.Sample* method), 455
- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 465
- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicData.SharedSamples* method), 456
- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicData-TimestampedSeq* method), 467
- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 459
- `__iter__()` (*rti.connexdds.PublicationBuiltinTopicData.ValidLoanedSamples* method), 460
- `__iter__()` (*rti.connexdds.PublisherSeq* method), 477
- `__iter__()` (*rti.connexdds.QosPolicyCountSeq* method), 479
- `__iter__()` (*rti.connexdds.Query* method), 485
- `__iter__()` (*rti.connexdds.ServiceRequest.ContentFiltered-TopicSeq* method), 526
- `__iter__()` (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 535
- `__iter__()` (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 546
- `__iter__()` (*rti.connexdds.ServiceRequest.LoanedSample* method), 548
- `__iter__()` (*rti.connexdds.ServiceRequest.LoanedSamples* method), 548
- `__iter__()` (*rti.connexdds.ServiceRequest.Sample* method), 551
- `__iter__()` (*rti.connexdds.ServiceRequestSeq* method), 561
- `__iter__()` (*rti.connexdds.ServiceRequest.SharedSamples* method), 552
- `__iter__()` (*rti.connexdds.ServiceRequestTimestampedSeq* method), 563
- `__iter__()` (*rti.connexdds.ServiceRequest.TopicSeq* method), 554
- `__iter__()` (*rti.connexdds.ServiceRequest.ValidLoanedSamples* method), 556
- `__iter__()` (*rti.connexdds.StringMap* method), 572
- `__iter__()` (*rti.connexdds.StringPairSeq* method), 573
- `__iter__()` (*rti.connexdds.StringSeq* method), 575
- `__iter__()` (*rti.connexdds.SubscriberSeq* method), 583
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 587
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 596
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 608
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicData.LoanedSample* method), 610
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples* method), 610
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicData.Sample* method), 614
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 623
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicData.SharedSamples* method), 614
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicData-TimestampedSeq* method), 626
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 617
- `__iter__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ValidLoanedSamples* method), 619
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 642
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq* method), 652
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq* method), 663
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicData.LoanedSample* method), 665
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicData.LoanedSamples* method), 665
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicData.Sample* method), 668
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicDataSeq* method), 677
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicData.SharedSamples* method), 669
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicData-TimestampedSeq* method), 679
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicData.TopicSeq* method), 672
- `__iter__()` (*rti.connexdds.TopicBuiltinTopicData.ValidLoanedSamples* method), 673
- `__iter__()` (*rti.connexdds.TopicData* method), 681
- `__iter__()` (*rti.connexdds.TopicSeq* method), 691
- `__iter__()` (*rti.connexdds.TransportInfoSeq* method), 700
- `__iter__()` (*rti.connexdds.TransportInfoVector* method), 702
- `__iter__()` (*rti.connexdds.TransportMulticastSeq* method), 707

- `__iter__` () (*rti.connexdds.TransportMulticastSettingsSeq* method), 710
- `__iter__` () (*rti.connexdds.TransportUnicastSettingsSeq* method), 714
- `__iter__` () (*rti.connexdds.TriggeredConditions* method), 716
- `__iter__` () (*rti.connexdds.Uint8Seq* method), 735
- `__iter__` () (*rti.connexdds.Uint16Seq* method), 727
- `__iter__` () (*rti.connexdds.Uint32Seq* method), 730
- `__iter__` () (*rti.connexdds.Uint64Seq* method), 732
- `__iter__` () (*rti.connexdds.UnionMemberSeq* method), 739
- `__iter__` () (*rti.connexdds.UserData* method), 741
- `__iter__` () (*rti.connexdds.WcharSeq* method), 750
- `__iter__` () (*rti.connexdds.WstringSeq* method), 757
- `__ixor__` () (*rti.connexdds.ActivityContextMask* method), 39
- `__ixor__` () (*rti.connexdds.CompressionIdMask* method), 71
- `__ixor__` () (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 152
- `__ixor__` () (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 155
- `__ixor__` () (*rti.connexdds.InstanceState* method), 312
- `__ixor__` () (*rti.connexdds.RtpsReservedPortKindMask* method), 502
- `__ixor__` () (*rti.connexdds.SampleFlag* method), 506
- `__ixor__` () (*rti.connexdds.SampleLostState* method), 511
- `__ixor__` () (*rti.connexdds.SampleRejectedState* method), 514
- `__ixor__` () (*rti.connexdds.SampleState* method), 517
- `__ixor__` () (*rti.connexdds.StatusMask* method), 568
- `__ixor__` () (*rti.connexdds.StreamKind* method), 570
- `__ixor__` () (*rti.connexdds.ThreadSettingsKindMask* method), 634
- `__ixor__` () (*rti.connexdds.TransportBuiltinMask* method), 694
- `__ixor__` () (*rti.connexdds.ViewState* method), 745
- `__le__` () (*rti.connexdds.AcknowledgmentKind* method), 37
- `__le__` () (*rti.connexdds.CdrPaddingKind* method), 62
- `__le__` () (*rti.connexdds.DataReaderInstanceRemovalKind* method), 93
- `__le__` () (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind* method), 137
- `__le__` () (*rti.connexdds.DestinationOrderKind* method), 144
- `__le__` () (*rti.connexdds.DestinationOrderScopeKind* method), 146
- `__le__` () (*rti.connexdds.DurabilityKind* method), 181
- `__le__` () (*rti.connexdds.Duration* method), 183
- `__le__` () (*rti.connexdds.DynamicDataEncapsulationKind* method), 247
- `__le__` () (*rti.connexdds.ExtensibilityKind* method), 270
- `__le__` () (*rti.connexdds.FlowControllerSchedulingPolicy* method), 284
- `__le__` () (*rti.connexdds.Guid* method), 287
- `__le__` () (*rti.connexdds.HistoryKind* method), 290
- `__le__` () (*rti.connexdds.IgnoredEntityReplacementKind* method), 307
- `__le__` () (*rti.connexdds.LivelinessKind* method), 330
- `__le__` () (*rti.connexdds.LocatorKind* method), 338
- `__le__` () (*rti.connexdds.LogCategory* method), 343
- `__le__` () (*rti.connexdds.OwnershipKind* method), 369
- `__le__` () (*rti.connexdds.PresentationAccessScopeKind* method), 418
- `__le__` () (*rti.connexdds.PrintFormat* method), 420
- `__le__` () (*rti.connexdds.PrintFormatKind* method), 422
- `__le__` () (*rti.connexdds.PublishModeKind* method), 472
- `__le__` () (*rti.connexdds.ReliabilityKind* method), 492
- `__le__` () (*rti.connexdds.RemoteParticipantPurgeKind* method), 495
- `__le__` () (*rti.connexdds.SequenceNumber* method), 519
- `__le__` () (*rti.connexdds.ServiceKind* method), 523
- `__le__` () (*rti.connexdds.ServiceRequestId* method), 560
- `__le__` () (*rti.connexdds.ThreadSettingsCpuRotationKind* method), 633
- `__le__` () (*rti.connexdds.Time* method), 637
- `__le__` () (*rti.connexdds.TopicQuerySelectionKind* method), 690
- `__le__` () (*rti.connexdds.TransportClassId* method), 698
- `__le__` () (*rti.connexdds.TransportMulticastKind* method), 704
- `__le__` () (*rti.connexdds.TypeConsistencyEnforcementKind* method), 719
- `__le__` () (*rti.connexdds.TypeKind* method), 725
- `__le__` () (*rti.connexdds.Verbosity* method), 744
- `__le__` () (*rti.connexdds.WireProtocolAutoKind* method), 754
- `__len__` () (*rti.connexdds.AnyDataReaderSeq* method), 44
- `__len__` () (*rti.connexdds.AnyDataWriterSeq* method), 47
- `__len__` () (*rti.connexdds.AnyTopicSeq* method), 50
- `__len__` () (*rti.connexdds.BoolSeq* method), 55
- `__len__` () (*rti.connexdds.ByteVector* method), 60
- `__len__` () (*rti.connexdds.ChannelSettingsSeq* method), 64
- `__len__` () (*rti.connexdds.CharSeq* method), 66
- `__len__` () (*rti.connexdds.ConditionSeq* method), 74
- `__len__` () (*rti.connexdds.ContentFilteredTopicSeq* method), 78
- `__len__` () (*rti.connexdds.CookieSeq* method), 81
- `__len__` () (*rti.connexdds.CookieVector* method), 82
- `__len__` () (*rti.connexdds.DataReader.LoanedSamples* method), 84
- `__len__` () (*rti.connexdds.DataReaderSeq* method), 108
- `__len__` () (*rti.connexdds.DataTag* method), 113
- `__len__` () (*rti.connexdds.DataWriterSeq* method), 138
- `__len__` () (*rti.connexdds.DomainParticipantSeq* method), 177
- `__len__` () (*rti.connexdds.DynamicData* method), 221
- `__len__` () (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 187
- `__len__` () (*rti.connexdds.DynamicData.DataReaderSeq* method), 196
- `__len__` () (*rti.connexdds.DynamicData.DataWriterSeq* method), 207
- `__len__` () (*rti.connexdds.DynamicData.FieldsView* method), 209
- `__len__` () (*rti.connexdds.DynamicData.ItemsView* method), 210
- `__len__` () (*rti.connexdds.DynamicData.LoanedSamples* method), 211
- `__len__` () (*rti.connexdds.DynamicDataSeq* method), 250

<code>__len__</code> () (<i>rti.connexdds.DynamicData.SharedSamples</i> method), 214	428
<code>__len__</code> () (<i>rti.connexdds.DynamicData.TimestampedSeq</i> method), 252	<code>__len__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq</i> method), 438
<code>__len__</code> () (<i>rti.connexdds.DynamicData.TopicSeq</i> method), 217	<code>__len__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq</i> method), 450
<code>__len__</code> () (<i>rti.connexdds.EndpointGroupSeq</i> method), 257	<code>__len__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.LoanedSamples</i> method), 452
<code>__len__</code> () (<i>rti.connexdds.EndpointGroupVector</i> method), 259	<code>__len__</code> () (<i>rti.connexdds.PublicationBuiltinTopicDataSeq</i> method), 465
<code>__len__</code> () (<i>rti.connexdds.EntitySeq</i> method), 262	<code>__len__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.SharedSamples</i> method), 456
<code>__len__</code> () (<i>rti.connexdds.EnumMemberSeq</i> method), 264	<code>__len__</code> () (<i>rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq</i> method), 467
<code>__len__</code> () (<i>rti.connexdds.Float32Seq</i> method), 275	<code>__len__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.TopicSeq</i> method), 459
<code>__len__</code> () (<i>rti.connexdds.Float64Seq</i> method), 278	<code>__len__</code> () (<i>rti.connexdds.PublisherSeq</i> method), 477
<code>__len__</code> () (<i>rti.connexdds.Float128Seq</i> method), 273	<code>__len__</code> () (<i>rti.connexdds.QosPolicyCountSeq</i> method), 479
<code>__len__</code> () (<i>rti.connexdds.Guid</i> method), 287	<code>__len__</code> () (<i>rti.connexdds.Query</i> method), 485
<code>__len__</code> () (<i>rti.connexdds.IAnyDataReaderSeq</i> method), 292	<code>__len__</code> () (<i>rti.connexdds.QueryCondition</i> method), 486
<code>__len__</code> () (<i>rti.connexdds.IAnyDataWriterSeq</i> method), 295	<code>__len__</code> () (<i>rti.connexdds.ServiceRequest.ContentFilteredTopicSeq</i> method), 526
<code>__len__</code> () (<i>rti.connexdds.IAnyTopicSeq</i> method), 298	<code>__len__</code> () (<i>rti.connexdds.ServiceRequest.DataReaderSeq</i> method), 535
<code>__len__</code> () (<i>rti.connexdds.IConditionSeq</i> method), 300	<code>__len__</code> () (<i>rti.connexdds.ServiceRequest.DataWriterSeq</i> method), 546
<code>__len__</code> () (<i>rti.connexdds.IEntitySeq</i> method), 303	<code>__len__</code> () (<i>rti.connexdds.ServiceRequest.LoanedSamples</i> method), 548
<code>__len__</code> () (<i>rti.connexdds.InstanceHandleSeq</i> method), 310	<code>__len__</code> () (<i>rti.connexdds.ServiceRequestSeq</i> method), 561
<code>__len__</code> () (<i>rti.connexdds.Int8Seq</i> method), 324	<code>__len__</code> () (<i>rti.connexdds.ServiceRequest.SharedSamples</i> method), 552
<code>__len__</code> () (<i>rti.connexdds.Int16Seq</i> method), 315	<code>__len__</code> () (<i>rti.connexdds.ServiceRequestTimestampedSeq</i> method), 563
<code>__len__</code> () (<i>rti.connexdds.Int32Seq</i> method), 318	<code>__len__</code> () (<i>rti.connexdds.ServiceRequest.TopicSeq</i> method), 554
<code>__len__</code> () (<i>rti.connexdds.Int32Vector</i> method), 320	<code>__len__</code> () (<i>rti.connexdds.StringMap</i> method), 572
<code>__len__</code> () (<i>rti.connexdds.Int64Seq</i> method), 321	<code>__len__</code> () (<i>rti.connexdds.StringPairSeq</i> method), 573
<code>__len__</code> () (<i>rti.connexdds.LocatorFilterElementSeq</i> method), 334	<code>__len__</code> () (<i>rti.connexdds.StringSeq</i> method), 575
<code>__len__</code> () (<i>rti.connexdds.LocatorSeq</i> method), 339	<code>__len__</code> () (<i>rti.connexdds.SubscriberSeq</i> method), 583
<code>__len__</code> () (<i>rti.connexdds.LocatorVector</i> method), 341	<code>__len__</code> () (<i>rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq</i> method), 587
<code>__len__</code> () (<i>rti.connexdds.MemberSeq</i> method), 348	<code>__len__</code> () (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq</i> method), 597
<code>__len__</code> () (<i>rti.connexdds.MonitoringMetricSelectionSeq</i> method), 354	<code>__len__</code> () (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq</i> method), 608
<code>__len__</code> () (<i>rti.connexdds.MulticastMappingSeq</i> method), 359	<code>__len__</code> () (<i>rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples</i> method), 610
<code>__len__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq</i> method), 372	<code>__len__</code> () (<i>rti.connexdds.SubscriptionBuiltinTopicDataSeq</i> method), 623
<code>__len__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq</i> method), 382	
<code>__len__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq</i> method), 393	
<code>__len__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples</i> method), 396	
<code>__len__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicDataSeq</i> method), 407	
<code>__len__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.SharedSamples</i> method), 399	
<code>__len__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq</i> method), 410	
<code>__len__</code> () (<i>rti.connexdds.ParticipantBuiltinTopicData.TopicSeq</i> method), 402	
<code>__len__</code> () (<i>rti.connexdds.Property</i> method), 424	
<code>__len__</code> () (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq</i> method),	

- `__len__()` (*rti.connextdds.SubscriptionBuiltinTopicData.SharedSamples* method), 614
- `__len__()` (*rti.connextdds.SubscriptionBuiltinTopicData.TimestampedSeq* method), 626
- `__len__()` (*rti.connextdds.SubscriptionBuiltinTopicData.TopicSeq* method), 617
- `__len__()` (*rti.connextdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 642
- `__len__()` (*rti.connextdds.TopicBuiltinTopicData.DataReaderSeq* method), 652
- `__len__()` (*rti.connextdds.TopicBuiltinTopicData.DataWriterSeq* method), 663
- `__len__()` (*rti.connextdds.TopicBuiltinTopicData.LoanedSamples* method), 665
- `__len__()` (*rti.connextdds.TopicBuiltinTopicDataSeq* method), 677
- `__len__()` (*rti.connextdds.TopicBuiltinTopicData.SharedSamples* method), 669
- `__len__()` (*rti.connextdds.TopicBuiltinTopicData.TimestampedSeq* method), 679
- `__len__()` (*rti.connextdds.TopicBuiltinTopicData.TopicSeq* method), 672
- `__len__()` (*rti.connextdds.TopicSeq* method), 691
- `__len__()` (*rti.connextdds.TransportInfoSeq* method), 700
- `__len__()` (*rti.connextdds.TransportInfoVector* method), 702
- `__len__()` (*rti.connextdds.TransportMulticastSeq* method), 707
- `__len__()` (*rti.connextdds.TransportMulticastSettingsSeq* method), 710
- `__len__()` (*rti.connextdds.TransportUnicastSettingsSeq* method), 714
- `__len__()` (*rti.connextdds.TriggeredConditions* method), 716
- `__len__()` (*rti.connextdds.Uint8Seq* method), 735
- `__len__()` (*rti.connextdds.Uint16Seq* method), 727
- `__len__()` (*rti.connextdds.Uint32Seq* method), 730
- `__len__()` (*rti.connextdds.Uint64Seq* method), 732
- `__len__()` (*rti.connextdds.UnionMemberSeq* method), 739
- `__len__()` (*rti.connextdds.WcharSeq* method), 750
- `__len__()` (*rti.connextdds.WstringSeq* method), 757
- `__long__()` (*rti.connextdds.EnumMember* method), 263
- `__lshift__()` (*rti.connextdds.ActivityContextMask* method), 39
- `__lshift__()` (*rti.connextdds.CompressionIdMask* method), 71
- `__lshift__()` (*rti.connextdds.DataReader* method), 86
- `__lshift__()` (*rti.connextdds.DataReaderQos* method), 97
- `__lshift__()` (*rti.connextdds.DataState* method), 110
- `__lshift__()` (*rti.connextdds.DataStateEx* method), 112
- `__lshift__()` (*rti.connextdds.DataWriter* method), 115
- `__lshift__()` (*rti.connextdds.DataWriterQos* method), 125
- `__lshift__()` (*rti.connextdds.DiscoveryConfigBuiltinChannelKindMask* method), 152
- `__lshift__()` (*rti.connextdds.DiscoveryConfigBuiltinPluginKindMask* method), 155
- `__lshift__()` (*rti.connextdds.DomainParticipant* method), 157
- `__lshift__()` (*rti.connextdds.DomainParticipantFactoryQos* method), 165
- `__lshift__()` (*rti.connextdds.DomainParticipantQos* method), 167
- `__lshift__()` (*rti.connextdds.DynamicData.DataReader* method), 190
- `__lshift__()` (*rti.connextdds.DynamicData.DataWriter* method), 198
- `__lshift__()` (*rti.connextdds.InstanceState* method), 312
- `__lshift__()` (*rti.connextdds.ParticipantBuiltinTopicData.DataReader* method), 376
- `__lshift__()` (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter* method), 384
- `__lshift__()` (*rti.connextdds.PublicationBuiltinTopicData.DataReader* method), 432
- `__lshift__()` (*rti.connextdds.PublicationBuiltinTopicData.DataWriter* method), 440
- `__lshift__()` (*rti.connextdds.Publisher* method), 472
- `__lshift__()` (*rti.connextdds.PublisherQos* method), 474
- `__lshift__()` (*rti.connextdds.RtpsReservedPortKindMask* method), 502
- `__lshift__()` (*rti.connextdds.SampleFlag* method), 506
- `__lshift__()` (*rti.connextdds.SampleLostState* method), 511
- `__lshift__()` (*rti.connextdds.SampleRejectedState* method), 514
- `__lshift__()` (*rti.connextdds.SampleState* method), 517
- `__lshift__()` (*rti.connextdds.ServiceRequest.DataReader* method), 529
- `__lshift__()` (*rti.connextdds.ServiceRequest.DataWriter* method), 537
- `__lshift__()` (*rti.connextdds.StatusMask* method), 568
- `__lshift__()` (*rti.connextdds.StreamKind* method), 570
- `__lshift__()` (*rti.connextdds.SubscriberQos* method), 580
- `__lshift__()` (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader* method), 591
- `__lshift__()` (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter* method), 599
- `__lshift__()` (*rti.connextdds.ThreadSettingsKindMask* method), 634
- `__lshift__()` (*rti.connextdds.TopicBuiltinTopicData.DataReader* method), 646
- `__lshift__()` (*rti.connextdds.TopicBuiltinTopicData.DataWriter* method), 654

- `__lshift__()` (*rti.connexdds.TopicQos method*), 682
- `__lshift__()` (*rti.connexdds.TransportBuiltinMask method*), 694
- `__lshift__()` (*rti.connexdds.ViewState method*), 745
- `__lt__()` (*rti.connexdds.AcknowledgmentKind method*), 37
- `__lt__()` (*rti.connexdds.CdrPaddingKind method*), 62
- `__lt__()` (*rti.connexdds.DataReaderInstanceRemovalKind method*), 94
- `__lt__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind method*), 137
- `__lt__()` (*rti.connexdds.DestinationOrderKind method*), 144
- `__lt__()` (*rti.connexdds.DestinationOrderScopeKind method*), 146
- `__lt__()` (*rti.connexdds.DurabilityKind method*), 181
- `__lt__()` (*rti.connexdds.Duration method*), 183
- `__lt__()` (*rti.connexdds.DynamicDataEncapsulationKind method*), 247
- `__lt__()` (*rti.connexdds.ExtensibilityKind method*), 270
- `__lt__()` (*rti.connexdds.FlowControllerSchedulingPolicy method*), 284
- `__lt__()` (*rti.connexdds.Guid method*), 287
- `__lt__()` (*rti.connexdds.HistoryKind method*), 290
- `__lt__()` (*rti.connexdds.IgnoredEntityReplacementKind method*), 307
- `__lt__()` (*rti.connexdds.LivelinessKind method*), 330
- `__lt__()` (*rti.connexdds.LocatorKind method*), 338
- `__lt__()` (*rti.connexdds.LogCategory method*), 343
- `__lt__()` (*rti.connexdds.OwnershipKind method*), 369
- `__lt__()` (*rti.connexdds.PresentationAccessScopeKind method*), 418
- `__lt__()` (*rti.connexdds.PrintFormat method*), 420
- `__lt__()` (*rti.connexdds.PrintFormatKind method*), 422
- `__lt__()` (*rti.connexdds.PublishModeKind method*), 472
- `__lt__()` (*rti.connexdds.ReliabilityKind method*), 492
- `__lt__()` (*rti.connexdds.RemoteParticipantPurgeKind method*), 495
- `__lt__()` (*rti.connexdds.SequenceNumber method*), 520
- `__lt__()` (*rti.connexdds.ServiceKind method*), 523
- `__lt__()` (*rti.connexdds.ServiceRequestId method*), 560
- `__lt__()` (*rti.connexdds.ThreadSettingsCpuRotationKind method*), 633
- `__lt__()` (*rti.connexdds.Time method*), 637
- `__lt__()` (*rti.connexdds.TopicQuerySelectionKind method*), 690
- `__lt__()` (*rti.connexdds.TransportClassId method*), 698
- `__lt__()` (*rti.connexdds.TransportMulticastKind method*), 704
- `__lt__()` (*rti.connexdds.TypeConsistencyEnforcementKind method*), 719
- `__lt__()` (*rti.connexdds.TypeKind method*), 725
- `__lt__()` (*rti.connexdds.Verbosity method*), 744
- `__lt__()` (*rti.connexdds.WireProtocolAutoKind method*), 755
- `__members__` (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind attribute*), 36
- `__members__` (*rti.connexdds.CdrPaddingKind.CdrPaddingKind attribute*), 61
- `__members__` (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind attribute*), 92
- `__members__` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 136
- `__members__` (*rti.connexdds.DestinationOrderKind.DestinationOrderKind attribute*), 143
- `__members__` (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind attribute*), 145
- `__members__` (*rti.connexdds.DurabilityKind.DurabilityKind attribute*), 180
- `__members__` (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind attribute*), 246
- `__members__` (*rti.connexdds.ExtensibilityKind.ExtensibilityKind attribute*), 269
- `__members__` (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy attribute*), 283
- `__members__` (*rti.connexdds.HistoryKind.HistoryKind attribute*), 289
- `__members__` (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind attribute*), 306
- `__members__` (*rti.connexdds.InstanceStateConsistencyKind attribute*), 314
- `__members__` (*rti.connexdds.LivelinessKind.LivelinessKind attribute*), 329
- `__members__` (*rti.connexdds.LocatorKind.LocatorKind attribute*), 336
- `__members__` (*rti.connexdds.LogCategory.LogCategory attribute*), 342
- `__members__` (*rti.connexdds.OwnershipKind.OwnershipKind attribute*), 368
- `__members__` (*rti.connexdds.PersistentJournalKind attribute*), 412
- `__members__` (*rti.connexdds.PersistentSynchronizationKind attribute*), 414
- `__members__` (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*), 417
- `__members__` (*rti.connexdds.PrintFormatKind.PrintFormatKind attribute*), 421
- `__members__` (*rti.connexdds.PrintFormat.PrintFormat attribute*), 419
- `__members__` (*rti.connexdds.PublishModeKind.PublishModeKind attribute*), 471
- `__members__` (*rti.connexdds.ReliabilityKind.ReliabilityKind attribute*), 491
- `__members__` (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind attribute*), 494

<code>__members__</code> (<i>rti.connexdds.ServiceKind.ServiceKind attribute</i>), 522	<code>__module__</code> (<i>rti.connexdds.BoolType attribute</i>), 56
<code>__members__</code> (<i>rti.connexdds.ServiceRequestId.ServiceRequestId attribute</i>), 559	<code>__module__</code> (<i>rti.connexdds.BuiltinProfiles attribute</i>), 56
<code>__members__</code> (<i>rti.connexdds.SyslogVerbosity attribute</i>), 629	<code>__module__</code> (<i>rti.connexdds.BuiltinTopicKey attribute</i>), 58
<code>__members__</code> (<i>rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind attribute</i>), 632	<code>__module__</code> (<i>rti.connexdds.BuiltinTopicReaderResourceLimits attribute</i>), 59
<code>__members__</code> (<i>rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind attribute</i>), 689	<code>__module__</code> (<i>rti.connexdds.ByteVector attribute</i>), 61
<code>__members__</code> (<i>rti.connexdds.TransportClassId.TransportClassId attribute</i>), 697	<code>__module__</code> (<i>rti.connexdds.CdrPaddingKind attribute</i>), 62
<code>__members__</code> (<i>rti.connexdds.TransportMulticastKind.TransportMulticastKind attribute</i>), 703	<code>__module__</code> (<i>rti.connexdds.CdrPaddingKind.CdrPaddingKind attribute</i>), 62
<code>__members__</code> (<i>rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind attribute</i>), 718	<code>__module__</code> (<i>rti.connexdds.ChannelSettings attribute</i>), 63
<code>__members__</code> (<i>rti.connexdds.TypeKind.TypeKind attribute</i>), 723	<code>__module__</code> (<i>rti.connexdds.ChannelSettingsSeq attribute</i>), 64
<code>__members__</code> (<i>rti.connexdds.Verbosity.Verbosity attribute</i>), 743	<code>__module__</code> (<i>rti.connexdds.CharSeq attribute</i>), 66
<code>__members__</code> (<i>rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind attribute</i>), 753	<code>__module__</code> (<i>rti.connexdds.CoherentAccess attribute</i>), 68
<code>__module__</code> (<i>rti.connexdds.AcknowledgmentInfo attribute</i>), 35	<code>__module__</code> (<i>rti.connexdds.CoherentSet attribute</i>), 68
<code>__module__</code> (<i>rti.connexdds.AcknowledgmentKind attribute</i>), 37	<code>__module__</code> (<i>rti.connexdds.CoherentSetInfo attribute</i>), 69
<code>__module__</code> (<i>rti.connexdds.AcknowledgmentKind.AcknowledgmentKind attribute</i>), 36	<code>__module__</code> (<i>rti.connexdds.CollectionType attribute</i>), 69
<code>__module__</code> (<i>rti.connexdds.AckResponseData attribute</i>), 34	<code>__module__</code> (<i>rti.connexdds.CompressionIdMask attribute</i>), 71
<code>__module__</code> (<i>rti.connexdds.ACTEnumMember attribute</i>), 31	<code>__module__</code> (<i>rti.connexdds.CompressionSettings attribute</i>), 73
<code>__module__</code> (<i>rti.connexdds.ActivityContextMask attribute</i>), 39	<code>__module__</code> (<i>rti.connexdds.Condition attribute</i>), 73
<code>__module__</code> (<i>rti.connexdds.ACTMember attribute</i>), 32	<code>__module__</code> (<i>rti.connexdds.ConditionSeq attribute</i>), 74
<code>__module__</code> (<i>rti.connexdds.ACTUnionMember attribute</i>), 33	<code>__module__</code> (<i>rti.connexdds.ContentFilterBase attribute</i>), 76
<code>__module__</code> (<i>rti.connexdds.AliasType attribute</i>), 41	<code>__module__</code> (<i>rti.connexdds.ContentFilteredTopic attribute</i>), 76
<code>__module__</code> (<i>rti.connexdds.AllocationSettings attribute</i>), 41	<code>__module__</code> (<i>rti.connexdds.ContentFilteredTopicSeq attribute</i>), 78
<code>__module__</code> (<i>rti.connexdds.AlreadyClosedError attribute</i>), 42	<code>__module__</code> (<i>rti.connexdds.ContentFilterProperty attribute</i>), 76
<code>__module__</code> (<i>rti.connexdds.AnyDataReader attribute</i>), 42	<code>__module__</code> (<i>rti.connexdds.Cookie attribute</i>), 80
<code>__module__</code> (<i>rti.connexdds.AnyDataReaderListener attribute</i>), 42	<code>__module__</code> (<i>rti.connexdds.CookieSeq attribute</i>), 81
<code>__module__</code> (<i>rti.connexdds.AnyDataReaderSeq attribute</i>), 44	<code>__module__</code> (<i>rti.connexdds.CookieVector attribute</i>), 83
<code>__module__</code> (<i>rti.connexdds.AnyDataWriter attribute</i>), 45	<code>__module__</code> (<i>rti.connexdds.Database attribute</i>), 140
<code>__module__</code> (<i>rti.connexdds.AnyDataWriterListener attribute</i>), 45	<code>__module__</code> (<i>rti.connexdds.DataReader attribute</i>), 86
<code>__module__</code> (<i>rti.connexdds.AnyDataWriterSeq attribute</i>), 47	<code>__module__</code> (<i>rti.connexdds.DataReaderCacheStatus attribute</i>), 90
<code>__module__</code> (<i>rti.connexdds.AnyTopic attribute</i>), 48	<code>__module__</code> (<i>rti.connexdds.DataReaderInstanceRemovalKind attribute</i>), 94
<code>__module__</code> (<i>rti.connexdds.AnyTopicListener attribute</i>), 48	<code>__module__</code> (<i>rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind attribute</i>), 92
<code>__module__</code> (<i>rti.connexdds.AnyTopicSeq attribute</i>), 50	<code>__module__</code> (<i>rti.connexdds.DataReaderListener attribute</i>), 94
<code>__module__</code> (<i>rti.connexdds.ArrayType attribute</i>), 51	<code>__module__</code> (<i>rti.connexdds.DataReader.LoanedSample attribute</i>), 83
<code>__module__</code> (<i>rti.connexdds.AsynchronousPublisher attribute</i>), 51	<code>__module__</code> (<i>rti.connexdds.DataReader.LoanedSamples attribute</i>), 84
<code>__module__</code> (<i>rti.connexdds.Availability attribute</i>), 52	<code>__module__</code> (<i>rti.connexdds.DataReader.Protocol attribute</i>), 95
<code>__module__</code> (<i>rti.connexdds.Batch attribute</i>), 53	<code>__module__</code> (<i>rti.connexdds.DataReader.ProtocolStatus attribute</i>), 96
<code>__module__</code> (<i>rti.connexdds.BoolSeq attribute</i>), 55	<code>__module__</code> (<i>rti.connexdds.DataReader.Qos attribute</i>), 100
	<code>__module__</code> (<i>rti.connexdds.DataReader.ResourceLimits attribute</i>), 104
	<code>__module__</code> (<i>rti.connexdds.DataReader.ResourceLimitsInstanceReplacementSettings attribute</i>), 107
	<code>__module__</code> (<i>rti.connexdds.DataReader.Selector attribute</i>), 84
	<code>__module__</code> (<i>rti.connexdds.DataReaderSeq attribute</i>), 108
	<code>__module__</code> (<i>rti.connexdds.DataRepresentation attribute</i>), 109
	<code>__module__</code> (<i>rti.connexdds.DataState attribute</i>), 111
	<code>__module__</code> (<i>rti.connexdds.DataStateEx attribute</i>), 112
	<code>__module__</code> (<i>rti.connexdds.DataTag attribute</i>), 113
	<code>__module__</code> (<i>rti.connexdds.DataWriter attribute</i>), 115

<code>__module__</code> (<i>rti.connexdds.DataWriterCacheStatus</i> attribute), 120	<code>__module__</code> (<i>rti.connexdds.DurabilityKind.DurabilityKind</i> attribute), 180
<code>__module__</code> (<i>rti.connexdds.DataWriterListener</i> attribute), 121	<code>__module__</code> (<i>rti.connexdds.DurabilityService</i> attribute), 182
<code>__module__</code> (<i>rti.connexdds.DataWriterProtocol</i> attribute), 122	<code>__module__</code> (<i>rti.connexdds.Duration</i> attribute), 183
<code>__module__</code> (<i>rti.connexdds.DataWriterProtocolStatus</i> attribute), 123	<code>__module__</code> (<i>rti.connexdds.DynamicData</i> attribute), 221
<code>__module__</code> (<i>rti.connexdds.DataWriterQos</i> attribute), 127	<code>__module__</code> (<i>rti.connexdds.DynamicData.ContentFilter</i> attribute), 184
<code>__module__</code> (<i>rti.connexdds.DataWriterResourceLimits</i> attribute), 132	<code>__module__</code> (<i>rti.connexdds.DynamicData.ContentFilteredTopic</i> attribute), 185
<code>__module__</code> (<i>rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind</i> attribute), 137	<code>__module__</code> (<i>rti.connexdds.DynamicData.ContentFilteredTopicSeq</i> attribute), 187
<code>__module__</code> (<i>rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind</i> attribute), 136	<code>__module__</code> (<i>rti.connexdds.DynamicData.DataReader</i> attribute), 190
<code>__module__</code> (<i>rti.connexdds.DataWriterSeq</i> attribute), 138	<code>__module__</code> (<i>rti.connexdds.DynamicData.DataReaderListener</i> attribute), 194
<code>__module__</code> (<i>rti.connexdds.DataWriterShmemRefTransferModeSettings</i> attribute), 140	<code>__module__</code> (<i>rti.connexdds.DynamicData.DataReader.Selector</i> attribute), 188
<code>__module__</code> (<i>rti.connexdds.DataWriterTransferMode</i> attribute), 140	<code>__module__</code> (<i>rti.connexdds.DynamicData.DataReaderSeq</i> attribute), 196
<code>__module__</code> (<i>rti.connexdds.Deadline</i> attribute), 141	<code>__module__</code> (<i>rti.connexdds.DynamicData.DataWriter</i> attribute), 198
<code>__module__</code> (<i>rti.connexdds.DestinationOrder</i> attribute), 142	<code>__module__</code> (<i>rti.connexdds.DynamicData.DataWriterListener</i> attribute), 205
<code>__module__</code> (<i>rti.connexdds.DestinationOrderKind</i> attribute), 144	<code>__module__</code> (<i>rti.connexdds.DynamicData.DataWriterSeq</i> attribute), 207
<code>__module__</code> (<i>rti.connexdds.DestinationOrderKind.DestinationOrderKind</i> attribute), 143	<code>__module__</code> (<i>rti.connexdds.DynamicData.EncapsulationKind</i> attribute), 247
<code>__module__</code> (<i>rti.connexdds.DestinationOrderScopeKind</i> attribute), 146	<code>__module__</code> (<i>rti.connexdds.DynamicData.EncapsulationKind.DynamicDataEncapsulationKind</i> attribute), 246
<code>__module__</code> (<i>rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind</i> attribute), 145	<code>__module__</code> (<i>rti.connexdds.DynamicData.FieldsIterator</i> attribute), 208
<code>__module__</code> (<i>rti.connexdds.Discovery</i> attribute), 147	<code>__module__</code> (<i>rti.connexdds.DynamicData.FieldsView</i> attribute), 209
<code>__module__</code> (<i>rti.connexdds.DiscoveryConfig</i> attribute), 148	<code>__module__</code> (<i>rti.connexdds.DynamicData.IndexIterator</i> attribute), 209
<code>__module__</code> (<i>rti.connexdds.DiscoveryConfigBuiltinChannelKindMask</i> attribute), 152	<code>__module__</code> (<i>rti.connexdds.DynamicData.Info</i> attribute), 248
<code>__module__</code> (<i>rti.connexdds.DiscoveryConfigBuiltinPluginKindMask</i> attribute), 155	<code>__module__</code> (<i>rti.connexdds.DynamicData.ItemsIterator</i> attribute), 209
<code>__module__</code> (<i>rti.connexdds.DomainParticipant</i> attribute), 158	<code>__module__</code> (<i>rti.connexdds.DynamicData.ItemsView</i> attribute), 210
<code>__module__</code> (<i>rti.connexdds.DomainParticipantConfigParams</i> attribute), 164	<code>__module__</code> (<i>rti.connexdds.DynamicData.ITopicDescription</i> attribute), 209
<code>__module__</code> (<i>rti.connexdds.DomainParticipantFactoryQos</i> attribute), 165	<code>__module__</code> (<i>rti.connexdds.DynamicData.LoanedSample</i> attribute), 210
<code>__module__</code> (<i>rti.connexdds.DomainParticipantListener</i> attribute), 166	<code>__module__</code> (<i>rti.connexdds.DynamicData.LoanedSamples</i> attribute), 211
<code>__module__</code> (<i>rti.connexdds.DomainParticipantProtocolStatus</i> attribute), 166	<code>__module__</code> (<i>rti.connexdds.DynamicData.MemberInfo</i> attribute), 248
<code>__module__</code> (<i>rti.connexdds.DomainParticipantQos</i> attribute), 168	<code>__module__</code> (<i>rti.connexdds.DynamicData.NoOpDataReaderListener</i> attribute), 211
<code>__module__</code> (<i>rti.connexdds.DomainParticipantResourceLimits</i> attribute), 171	<code>__module__</code> (<i>rti.connexdds.DynamicData.NoOpDataWriterListener</i> attribute), 212
<code>__module__</code> (<i>rti.connexdds.DomainParticipantSeq</i> attribute), 177	
<code>__module__</code> (<i>rti.connexdds.Durability</i> attribute), 178	
<code>__module__</code> (<i>rti.connexdds.DurabilityKind</i> attribute), 181	

- `__module__` (*rti.connexdds.DynamicData.NoOpTopicListener attribute*), 213
- `__module__` (*rti.connexdds.DynamicData.Property attribute*), 249
- `__module__` (*rti.connexdds.DynamicData.Sample attribute*), 214
- `__module__` (*rti.connexdds.DynamicData.Seq attribute*), 250
- `__module__` (*rti.connexdds.DynamicData.SharedSamples attribute*), 214
- `__module__` (*rti.connexdds.DynamicData.TimestampedSeq attribute*), 253
- `__module__` (*rti.connexdds.DynamicData.Topic attribute*), 215
- `__module__` (*rti.connexdds.DynamicData.TopicDescription attribute*), 216
- `__module__` (*rti.connexdds.DynamicData.TopicListener attribute*), 216
- `__module__` (*rti.connexdds.DynamicData.TopicSeq attribute*), 217
- `__module__` (*rti.connexdds.DynamicData.TopicTypeSupport attribute*), 218
- `__module__` (*rti.connexdds.DynamicDataTypeSerializationProperty attribute*), 254
- `__module__` (*rti.connexdds.DynamicData.ValidLoanedSamples attribute*), 219
- `__module__` (*rti.connexdds.DynamicData.WriterContentFilter attribute*), 219
- `__module__` (*rti.connexdds.DynamicData.WriterContentFilterHelper attribute*), 220
- `__module__` (*rti.connexdds.DynamicType attribute*), 255
- `__module__` (*rti.connexdds.DynamicTypePrintFormatProperty attribute*), 256
- `__module__` (*rti.connexdds.EndpointGroup attribute*), 256
- `__module__` (*rti.connexdds.EndpointGroupSeq attribute*), 257
- `__module__` (*rti.connexdds.EndpointGroupVector attribute*), 259
- `__module__` (*rti.connexdds.Entity attribute*), 260
- `__module__` (*rti.connexdds.EntityFactory attribute*), 260
- `__module__` (*rti.connexdds.EntityName attribute*), 260
- `__module__` (*rti.connexdds.EntitySeq attribute*), 262
- `__module__` (*rti.connexdds.EnumMember attribute*), 263
- `__module__` (*rti.connexdds.EnumMemberSeq attribute*), 264
- `__module__` (*rti.connexdds.EnumType attribute*), 266
- `__module__` (*rti.connexdds.Error attribute*), 266
- `__module__` (*rti.connexdds.Event attribute*), 267
- `__module__` (*rti.connexdds.EventCount32 attribute*), 267
- `__module__` (*rti.connexdds.EventCount64 attribute*), 267
- `__module__` (*rti.connexdds.Exception attribute*), 268
- `__module__` (*rti.connexdds.ExclusiveArea attribute*), 268
- `__module__` (*rti.connexdds.ExpressionProperty attribute*), 269
- `__module__` (*rti.connexdds.ExtensibilityKind attribute*), 270
- `__module__` (*rti.connexdds.ExtensibilityKind.ExtensibilityKind attribute*), 270
- `__module__` (*rti.connexdds.Filter attribute*), 271
- `__module__` (*rti.connexdds.FilterSampleInfo attribute*), 272
- `__module__` (*rti.connexdds.Float32Seq attribute*), 275
- `__module__` (*rti.connexdds.Float32Type attribute*), 277
- `__module__` (*rti.connexdds.Float64Seq attribute*), 278
- `__module__` (*rti.connexdds.Float64Type attribute*), 279
- `__module__` (*rti.connexdds.Float128Seq attribute*), 273
- `__module__` (*rti.connexdds.Float128Type attribute*), 274
- `__module__` (*rti.connexdds.FlowController attribute*), 280
- `__module__` (*rti.connexdds.FlowControllerProperty attribute*), 281
- `__module__` (*rti.connexdds.FlowControllerSchedulingPolicy attribute*), 284
- `__module__` (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy attribute*), 283
- `__module__` (*rti.connexdds.FlowControllerTokenBucketProperty attribute*), 285
- `__module__` (*rti.connexdds.GenerationCount attribute*), 285
- `__module__` (*rti.connexdds.GroupData attribute*), 286
- `__module__` (*rti.connexdds.GuardCondition attribute*), 286
- `__module__` (*rti.connexdds.Guid attribute*), 287
- `__module__` (*rti.connexdds.History attribute*), 288
- `__module__` (*rti.connexdds.HistoryKind attribute*), 290
- `__module__` (*rti.connexdds.HistoryKind.HistoryKind attribute*), 289
- `__module__` (*rti.connexdds.IAnyDataReader attribute*), 290
- `__module__` (*rti.connexdds.IAnyDataReaderSeq attribute*), 292
- `__module__` (*rti.connexdds.IAnyDataWriter attribute*), 293
- `__module__` (*rti.connexdds.IAnyDataWriterSeq attribute*), 295
- `__module__` (*rti.connexdds.IAnyTopic attribute*), 296
- `__module__` (*rti.connexdds.IAnyTopicSeq attribute*), 298
- `__module__` (*rti.connexdds.ICondition attribute*), 299
- `__module__` (*rti.connexdds.IConditionSeq attribute*), 300
- `__module__` (*rti.connexdds.IDataReader attribute*), 301
- `__module__` (*rti.connexdds.IEntity attribute*), 302
- `__module__` (*rti.connexdds.IEntitySeq attribute*), 303
- `__module__` (*rti.connexdds.IgnoredEntityReplacementKind attribute*), 307
- `__module__` (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind attribute*), 306
- `__module__` (*rti.connexdds.IllegalOperationError attribute*), 308
- `__module__` (*rti.connexdds.ImmutablePolicyError attribute*), 308
- `__module__` (*rti.connexdds.InconsistentPolicyError attribute*), 308
- `__module__` (*rti.connexdds.InconsistentTopicStatus attribute*), 308
- `__module__` (*rti.connexdds.InstanceHandle attribute*), 308
- `__module__` (*rti.connexdds.InstanceHandleSeq attribute*), 310
- `__module__` (*rti.connexdds.InstanceState attribute*), 312
- `__module__` (*rti.connexdds.InstanceStateConsistencyKind attribute*), 314
- `__module__` (*rti.connexdds.Int8Seq attribute*), 324
- `__module__` (*rti.connexdds.Int8Type attribute*), 325
- `__module__` (*rti.connexdds.Int16Seq attribute*), 315
- `__module__` (*rti.connexdds.Int16Type attribute*), 317

- `__module__` (*rti.connexdds.Int32Seq* attribute), 318
- `__module__` (*rti.connexdds.Int32Type* attribute), 319
- `__module__` (*rti.connexdds.Int32Vector* attribute), 320
- `__module__` (*rti.connexdds.Int64Seq* attribute), 321
- `__module__` (*rti.connexdds.Int64Type* attribute), 323
- `__module__` (*rti.connexdds.InvalidArgumentError* attribute), 325
- `__module__` (*rti.connexdds.InvalidDowncastError* attribute), 326
- `__module__` (*rti.connexdds.InvalidLocalIdentityAdvanceNoticeStatus* attribute), 326
- `__module__` (*rti.connexdds.IReadCondition* attribute), 305
- `__module__` (*rti.connexdds.ITopicDescription* attribute), 305
- `__module__` (*rti.connexdds.LatencyBudget* attribute), 326
- `__module__` (*rti.connexdds.Lifespan* attribute), 327
- `__module__` (*rti.connexdds.Liveliness* attribute), 327
- `__module__` (*rti.connexdds.LivelinessChangedStatus* attribute), 328
- `__module__` (*rti.connexdds.LivelinessKind* attribute), 330
- `__module__` (*rti.connexdds.LivelinessKind.LivelinessKind* attribute), 329
- `__module__` (*rti.connexdds.LivelinessLostStatus* attribute), 330
- `__module__` (*rti.connexdds.LoanedDynamicData* attribute), 331
- `__module__` (*rti.connexdds.Locator* attribute), 331
- `__module__` (*rti.connexdds.LocatorFilter* attribute), 332
- `__module__` (*rti.connexdds.LocatorFilterElement* attribute), 332
- `__module__` (*rti.connexdds.LocatorFilterElementSeq* attribute), 334
- `__module__` (*rti.connexdds.LocatorKind* attribute), 338
- `__module__` (*rti.connexdds.LocatorKind.LocatorKind* attribute), 337
- `__module__` (*rti.connexdds.LocatorSeq* attribute), 339
- `__module__` (*rti.connexdds.LocatorVector* attribute), 341
- `__module__` (*rti.connexdds.LogCategory* attribute), 343
- `__module__` (*rti.connexdds.LogCategory.LogCategory* attribute), 342
- `__module__` (*rti.connexdds.Logger* attribute), 344
- `__module__` (*rti.connexdds.LongDouble* attribute), 345
- `__module__` (*rti.connexdds.Member* attribute), 346
- `__module__` (*rti.connexdds.MemberSeq* attribute), 348
- `__module__` (*rti.connexdds.Monitoring* attribute), 349
- `__module__` (*rti.connexdds.MonitoringDedicatedParticipantSettings* attribute), 350
- `__module__` (*rti.connexdds.MonitoringDistributionSettings* attribute), 350
- `__module__` (*rti.connexdds.MonitoringEventDistributionSettings* attribute), 351
- `__module__` (*rti.connexdds.MonitoringLoggingDistributionSettings* attribute), 352
- `__module__` (*rti.connexdds.MonitoringLoggingForwardingSettings* attribute), 352
- `__module__` (*rti.connexdds.MonitoringMetricSelection* attribute), 353
- `__module__` (*rti.connexdds.MonitoringMetricSelectionSeq* attribute), 354
- `__module__` (*rti.connexdds.MonitoringPeriodicDistributionSettings* attribute), 356
- `__module__` (*rti.connexdds.MonitoringTelemetryData* attribute), 356
- `__module__` (*rti.connexdds.MulticastMapping* attribute), 358
- `__module__` (*rti.connexdds.MulticastMappingSeq* attribute), 359
- `__module__` (*rti.connexdds.MultiChannel* attribute), 357
- `__module__` (*rti.connexdds.NoOpAnyDataReaderListener* attribute), 360
- `__module__` (*rti.connexdds.NoOpAnyDataWriterListener* attribute), 361
- `__module__` (*rti.connexdds.NoOpAnyTopicListener* attribute), 362
- `__module__` (*rti.connexdds.NoOpDataReaderListener* attribute), 362
- `__module__` (*rti.connexdds.NoOpDataWriterListener* attribute), 363
- `__module__` (*rti.connexdds.NoOpDomainParticipantListener* attribute), 364
- `__module__` (*rti.connexdds.NoOpPublisherListener* attribute), 365
- `__module__` (*rti.connexdds.NoOpSubscriberListener* attribute), 365
- `__module__` (*rti.connexdds.NoOpTopicListener* attribute), 365
- `__module__` (*rti.connexdds.NotAllowedBySecurityError* attribute), 365
- `__module__` (*rti.connexdds.NotEnabledError* attribute), 365
- `__module__` (*rti.connexdds.NullReferenceError* attribute), 365
- `__module__` (*rti.connexdds.OfferedDeadlineMissedStatus* attribute), 366
- `__module__` (*rti.connexdds.OfferedIncompatibleQosStatus* attribute), 366
- `__module__` (*rti.connexdds.OutOfResourcesError* attribute), 366
- `__module__` (*rti.connexdds.Ownership* attribute), 367
- `__module__` (*rti.connexdds.OwnershipKind* attribute), 369
- `__module__` (*rti.connexdds.OwnershipKind.OwnershipKind* attribute), 368
- `__module__` (*rti.connexdds.OwnershipStrength* attribute), 369
- `__module__` (*rti.connexdds.ParticipantBuiltinTopicData* attribute), 405
- `__module__` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilter* attribute), 370
- `__module__` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic* attribute), 370
- `__module__` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* attribute), 372
- `__module__` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* attribute), 376

<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener</i> attribute), 380	<code>Data.ValidLoanedSamples</code> attribute), 404
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector</i> attribute), 374	<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter</i> attribute), 404
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq</i> attribute), 382	<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilterHelper</i> attribute), 405
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.DataWriter</i> attribute), 384	<code>__module__</code> (<i>rti.connexdds.Partition</i> attribute), 411
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.DataWriterListener</i> attribute), 391	<code>__module__</code> (<i>rti.connexdds.PersistentJournalKind</i> attribute), 412
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq</i> attribute), 393	<code>__module__</code> (<i>rti.connexdds.PersistentStorageSettings</i> attribute), 413
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.ITopicDescription</i> attribute), 395	<code>__module__</code> (<i>rti.connexdds.PersistentSynchronizationKind</i> attribute), 414
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.LoanedSample</i> attribute), 395	<code>__module__</code> (<i>rti.connexdds.PreconditionNotMetError</i> attribute), 415
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples</i> attribute), 396	<code>__module__</code> (<i>rti.connexdds.Presentation</i> attribute), 415
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener</i> attribute), 396	<code>__module__</code> (<i>rti.connexdds.PresentationAccessScopeKind</i> attribute), 418
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.NoOpDataWriterListener</i> attribute), 397	<code>__module__</code> (<i>rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind</i> attribute), 417
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.NoOpTopicListener</i> attribute), 398	<code>__module__</code> (<i>rti.connexdds.PrintFormat</i> attribute), 420
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.Sample</i> attribute), 399	<code>__module__</code> (<i>rti.connexdds.PrintFormatKind</i> attribute), 422
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicDataSeq</i> attribute), 407	<code>__module__</code> (<i>rti.connexdds.PrintFormatKind.PrintFormatKind</i> attribute), 421
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.SharedSamples</i> attribute), 399	<code>__module__</code> (<i>rti.connexdds.PrintFormat.PrintFormat</i> attribute), 419
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq</i> attribute), 410	<code>__module__</code> (<i>rti.connexdds.PrintFormatProperty</i> attribute), 422
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.Topic</i> attribute), 400	<code>__module__</code> (<i>rti.connexdds.ProductVersion</i> attribute), 423
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.TopicDescription</i> attribute), 401	<code>__module__</code> (<i>rti.connexdds.Property</i> attribute), 424
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.TopicListener</i> attribute), 401	<code>__module__</code> (<i>rti.connexdds.ProtocolVersion</i> attribute), 425
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopicData.TopicSeq</i> attribute), 402	<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData</i> attribute), 462
<code>__module__</code> (<i>rti.connexdds.ParticipantBuiltinTopic-</i>	<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilter</i> attribute), 426
	<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic</i> attribute), 427
	<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq</i> attribute), 428
	<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.DataReader</i> attribute), 432
	<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.DataReaderListener</i> attribute), 436
	<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector</i> attribute), 430
	<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq</i> attribute), 438
	<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-</i>

<i>Data.Data Writer attribute</i>),	<code>__module__</code> (<i>rti.connexdds.Publisher attribute</i>), 472
441	<code>__module__</code> (<i>rti.connexdds.PublisherListener attribute</i>), 474
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.Data WriterListener attribute</i>),	<code>__module__</code> (<i>rti.connexdds.PublisherQos attribute</i>), 475
447	<code>__module__</code> (<i>rti.connexdds.PublisherSeq attribute</i>), 477
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.Data WriterSeq attribute</i>),	<code>__module__</code> (<i>rti.connexdds.PublishMode attribute</i>), 469
450	<code>__module__</code> (<i>rti.connexdds.PublishModeKind attribute</i>), 472
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.ITopicDescription attribute</i>),	<code>__module__</code> (<i>rti.connexdds.PublishModeKind.PublishModeKind attribute</i>), 471
451	<code>__module__</code> (<i>rti.connexdds.QosPolicyCount attribute</i>), 478
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.LoanedSample attribute</i>),	<code>__module__</code> (<i>rti.connexdds.QosPolicyCountSeq attribute</i>), 479
451	<code>__module__</code> (<i>rti.connexdds.QosPrintFormat attribute</i>), 481
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.LoanedSamples attribute</i>),	<code>__module__</code> (<i>rti.connexdds.QosProvider attribute</i>), 481
452	<code>__module__</code> (<i>rti.connexdds.QosProviderParams attribute</i>),
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.NoOpDataReaderListener attribute</i>),	484
452	<code>__module__</code> (<i>rti.connexdds.Query attribute</i>), 485
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.NoOpDataWriterListener attribute</i>),	<code>__module__</code> (<i>rti.connexdds.QueryCondition attribute</i>), 486
452	<code>__module__</code> (<i>rti.connexdds.Rank attribute</i>), 487
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.NoOpTopicListener attribute</i>),	<code>__module__</code> (<i>rti.connexdds.ReadCondition attribute</i>), 487
453	<code>__module__</code> (<i>rti.connexdds.ReaderDataLifecycle attribute</i>),
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.NoOpTopicListener attribute</i>),	488
455	<code>__module__</code> (<i>rti.connexdds.ReceiverPool attribute</i>), 489
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.Sample attribute</i>), 455	<code>__module__</code> (<i>rti.connexdds.Reliability attribute</i>), 490
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicDataSeq attribute</i>), 465	<code>__module__</code> (<i>rti.connexdds.ReliabilityKind attribute</i>), 492
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.SharedSamples attribute</i>),	<code>__module__</code> (<i>rti.connexdds.ReliabilityKind.ReliabilityKind attribute</i>), 491
456	<code>__module__</code> (<i>rti.connexdds.ReliableReaderActivityChangedStatus attribute</i>), 492
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData-TimestampedSeq attribute</i>),	<code>__module__</code> (<i>rti.connexdds.ReliableWriterCacheChangedStatus attribute</i>), 492
467	<code>__module__</code> (<i>rti.connexdds.RemoteParticipantPurgeKind attribute</i>), 495
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.Topic attribute</i>), 456	<code>__module__</code> (<i>rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind attribute</i>),
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.TopicDescription attribute</i>),	494
457	<code>__module__</code> (<i>rti.connexdds.RequestedDeadlineMissedStatus attribute</i>), 496
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.TopicListener attribute</i>),	<code>__module__</code> (<i>rti.connexdds.RequestedIncompatibleQosStatus attribute</i>), 496
458	<code>__module__</code> (<i>rti.connexdds.ResourceLimits attribute</i>), 497
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopicData.TopicSeq attribute</i>), 459	<code>__module__</code> (<i>rti.connexdds.RtpsReliableReaderProtocol attribute</i>), 497
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.ValidLoanedSamples attribute</i>),	<code>__module__</code> (<i>rti.connexdds.RtpsReliableWriterProtocol attribute</i>), 498
460	<code>__module__</code> (<i>rti.connexdds.RtpsReservedPortKindMask attribute</i>), 502
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.WriterContentFilter attribute</i>),	<code>__module__</code> (<i>rti.connexdds.RtpsWellKnownPorts attribute</i>),
460	504
<code>__module__</code> (<i>rti.connexdds.PublicationBuiltinTopic-Data.WriterContentFilterHelper attribute</i>),	<code>__module__</code> (<i>rti.connexdds.SampleFlag attribute</i>), 506
461	<code>__module__</code> (<i>rti.connexdds.SampleIdentity attribute</i>), 507
<code>__module__</code> (<i>rti.connexdds.PublicationMatchedStatus attribute</i>), 469	<code>__module__</code> (<i>rti.connexdds.SampleInfo attribute</i>), 508
	<code>__module__</code> (<i>rti.connexdds.SampleLostState attribute</i>), 511
	<code>__module__</code> (<i>rti.connexdds.SampleLostStatus attribute</i>), 513
	<code>__module__</code> (<i>rti.connexdds.SampleRejectedState attribute</i>),
	514
	<code>__module__</code> (<i>rti.connexdds.SampleRejectedStatus attribute</i>),
	516

<code>__module__</code> (<i>rti.connexdds.SampleState</i> attribute), 517	<code>__module__</code> (<i>rti.connexdds.ServiceRequest.Topic</i> attribute), 552
<code>__module__</code> (<i>rti.connexdds.SequenceNumber</i> attribute), 520	<code>__module__</code> (<i>rti.connexdds.ServiceRequest.TopicDescription</i> attribute), 553
<code>__module__</code> (<i>rti.connexdds.SequenceType</i> attribute), 520	<code>__module__</code> (<i>rti.connexdds.ServiceRequest.TopicListener</i> attribute), 553
<code>__module__</code> (<i>rti.connexdds.Service</i> attribute), 521	<code>__module__</code> (<i>rti.connexdds.ServiceRequest.TopicSeq</i> attribute), 554
<code>__module__</code> (<i>rti.connexdds.ServiceKind</i> attribute), 523	<code>__module__</code> (<i>rti.connexdds.ServiceRequest.ValidLoanedSamples</i> attribute), 556
<code>__module__</code> (<i>rti.connexdds.ServiceKind.ServiceKind</i> attribute), 522	<code>__module__</code> (<i>rti.connexdds.ServiceRequest.WriterContentFilter</i> attribute), 556
<code>__module__</code> (<i>rti.connexdds.ServiceRequest</i> attribute), 557	<code>__module__</code> (<i>rti.connexdds.ServiceRequest.WriterContentFilterHelper</i> attribute), 557
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.AcceptedStatus</i> attribute), 558	<code>__module__</code> (<i>rti.connexdds.StatusCondition</i> attribute), 565
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.ContentFilter</i> attribute), 524	<code>__module__</code> (<i>rti.connexdds.StatusMask</i> attribute), 568
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.ContentFilteredTopic</i> attribute), 524	<code>__module__</code> (<i>rti.connexdds.StreamKind</i> attribute), 570
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.ContentFiltered-TopicSeq</i> attribute), 526	<code>__module__</code> (<i>rti.connexdds.StringMap</i> attribute), 572
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.DataReader</i> attribute), 530	<code>__module__</code> (<i>rti.connexdds.StringPairSeq</i> attribute), 573
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.DataReaderListener</i> attribute), 533	<code>__module__</code> (<i>rti.connexdds.StringSeq</i> attribute), 575
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.DataReader.Selector</i> attribute), 527	<code>__module__</code> (<i>rti.connexdds.StringType</i> attribute), 577
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.DataReaderSeq</i> attribute), 535	<code>__module__</code> (<i>rti.connexdds.StructType</i> attribute), 578
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.DataWriter</i> attribute), 537	<code>__module__</code> (<i>rti.connexdds.Subscriber</i> attribute), 579
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.DataWriterListener</i> attribute), 544	<code>__module__</code> (<i>rti.connexdds.SubscriberListener</i> attribute), 580
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.DataWriterSeq</i> attribute), 546	<code>__module__</code> (<i>rti.connexdds.SubscriberQos</i> attribute), 581
<code>__module__</code> (<i>rti.connexdds.ServiceRequestId</i> attribute), 560	<code>__module__</code> (<i>rti.connexdds.SubscriberSeq</i> attribute), 583
<code>__module__</code> (<i>rti.connexdds.ServiceRequestId.ServiceRequestId</i> attribute), 559	<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData</i> attribute), 620
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.ITopicDescription</i> attribute), 547	<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.ContentFilter</i> attribute), 584
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.LoanedSample</i> attribute), 548	<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic</i> attribute), 585
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.LoanedSamples</i> attribute), 548	<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq</i> attribute), 587
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.NoOpDataReaderListener</i> attribute), 549	<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataReader</i> attribute), 591
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.NoOp-DataWriterListener</i> attribute), 549	<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataReaderListener</i> attribute), 594
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.NoOpTopicListener</i> attribute), 551	<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector</i> attribute), 588
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.Sample</i> attribute), 551	<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq</i> attribute), 597
<code>__module__</code> (<i>rti.connexdds.ServiceRequestSeq</i> attribute), 561	<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataWriter</i> attribute), 599
<code>__module__</code> (<i>rti.connexdds.ServiceRequest.SharedSamples</i> attribute), 552	<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener</i> attribute), 606
<code>__module__</code> (<i>rti.connexdds.ServiceRequestTimestampedSeq</i> attribute), 564	

<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq</i> attribute), 608	<code>__module__</code> (<i>rti.connexdds.ThreadContext</i> attribute), 630
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.ITopicDescription</i> attribute), 609	<code>__module__</code> (<i>rti.connexdds.ThreadSettings</i> attribute), 631
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.LoanedSample</i> attribute), 610	<code>__module__</code> (<i>rti.connexdds.ThreadSettingsCpuRotationKind</i> attribute), 633
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples</i> attribute), 611	<code>__module__</code> (<i>rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind</i> attribute), 632
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener</i> attribute), 611	<code>__module__</code> (<i>rti.connexdds.ThreadSettingsKindMask</i> attribute), 634
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener</i> attribute), 612	<code>__module__</code> (<i>rti.connexdds.Time</i> attribute), 637
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.NoOpTopicListener</i> attribute), 613	<code>__module__</code> (<i>rti.connexdds.TimeBasedFilter</i> attribute), 638
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.Sample</i> attribute), 614	<code>__module__</code> (<i>rti.connexdds.TimeoutError</i> attribute), 638
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicDataSeq</i> attribute), 623	<code>__module__</code> (<i>rti.connexdds.Topic</i> attribute), 639
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.SharedSamples</i> attribute), 614	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData</i> attribute), 674
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.TimestampedSeq</i> attribute), 626	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.ContentFilter</i> attribute), 640
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.Topic</i> attribute), 615	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic</i> attribute), 640
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription</i> attribute), 616	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq</i> attribute), 642
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.TopicListener</i> attribute), 616	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.DataReader</i> attribute), 646
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq</i> attribute), 617	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.DataReaderListener</i> attribute), 650
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.ValidLoanedSamples</i> attribute), 619	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.DataReader.Selector</i> attribute), 644
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter</i> attribute), 619	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.DataReaderSeq</i> attribute), 652
<code>__module__</code> (<i>rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper</i> attribute), 620	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.DataWriter</i> attribute), 654
<code>__module__</code> (<i>rti.connexdds.SubscriptionMatchedStatus</i> attribute), 627	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.DataWriterListener</i> attribute), 661
<code>__module__</code> (<i>rti.connexdds.SuspendedPublication</i> attribute), 628	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.DataWriterSeq</i> attribute), 663
<code>__module__</code> (<i>rti.connexdds.SyslogVerbosity</i> attribute), 629	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.ITopicDescription</i> attribute), 664
<code>__module__</code> (<i>rti.connexdds.SystemResourceLimits</i> attribute), 630	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.LoanedSample</i> attribute), 665
	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.LoanedSamples</i> attribute), 665
	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener</i> attribute), 665
	<code>__module__</code> (<i>rti.connexdds.TopicBuiltinTopicData.NoOp-</i>

- Data WriterListener attribute*), 666
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.NoOpTopicListener attribute*), 668
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.Sample attribute*), 668
- `__module__` (*rti.connexdds.TopicBuiltinTopicDataSeq attribute*), 677
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.SharedSamples attribute*), 669
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.TimestampedSeq attribute*), 679
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.Topic attribute*), 669
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.TopicDescription attribute*), 670
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.TopicListener attribute*), 671
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.TopicSeq attribute*), 672
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.ValidLoanedSamples attribute*), 673
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilter attribute*), 673
- `__module__` (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilterHelper attribute*), 674
- `__module__` (*rti.connexdds.TopicData attribute*), 681
- `__module__` (*rti.connexdds.TopicDescription attribute*), 681
- `__module__` (*rti.connexdds.TopicListener attribute*), 682
- `__module__` (*rti.connexdds.TopicQos attribute*), 683
- `__module__` (*rti.connexdds.TopicQuery attribute*), 686
- `__module__` (*rti.connexdds.TopicQueryData attribute*), 687
- `__module__` (*rti.connexdds.TopicQueryDispatch attribute*), 687
- `__module__` (*rti.connexdds.TopicQuerySelection attribute*), 688
- `__module__` (*rti.connexdds.TopicQuerySelectionKind attribute*), 690
- `__module__` (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind attribute*), 689
- `__module__` (*rti.connexdds.TopicSeq attribute*), 691
- `__module__` (*rti.connexdds.TransportBuiltin attribute*), 692
- `__module__` (*rti.connexdds.TransportBuiltinMask attribute*), 694
- `__module__` (*rti.connexdds.TransportClassId attribute*), 699
- `__module__` (*rti.connexdds.TransportClassId.TransportClassId attribute*), 698
- `__module__` (*rti.connexdds.TransportInfo attribute*), 699
- `__module__` (*rti.connexdds.TransportInfoSeq attribute*), 700
- `__module__` (*rti.connexdds.TransportInfoVector attribute*), 702
- `__module__` (*rti.connexdds.TransportMulticast attribute*), 703
- `__module__` (*rti.connexdds.TransportMulticastKind attribute*), 704
- `__module__` (*rti.connexdds.TransportMulticastKind.TransportMulticastKind attribute*), 703
- `__module__` (*rti.connexdds.TransportMulticastMapping attribute*), 705
- `__module__` (*rti.connexdds.TransportMulticastMappingFunction attribute*), 706
- `__module__` (*rti.connexdds.TransportMulticastSeq attribute*), 707
- `__module__` (*rti.connexdds.TransportMulticastSettings attribute*), 708
- `__module__` (*rti.connexdds.TransportMulticastSettingsSeq attribute*), 710
- `__module__` (*rti.connexdds.TransportPriority attribute*), 711
- `__module__` (*rti.connexdds.TransportSelection attribute*), 712
- `__module__` (*rti.connexdds.TransportUnicast attribute*), 712
- `__module__` (*rti.connexdds.TransportUnicastSettings attribute*), 713
- `__module__` (*rti.connexdds.TransportUnicastSettingsSeq attribute*), 714
- `__module__` (*rti.connexdds.TriggeredConditions attribute*), 716
- `__module__` (*rti.connexdds.TriggeredConditionsIterator attribute*), 716
- `__module__` (*rti.connexdds.TypeConsistencyEnforcement attribute*), 716
- `__module__` (*rti.connexdds.TypeConsistencyEnforcementKind attribute*), 719
- `__module__` (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind attribute*), 718
- `__module__` (*rti.connexdds.TypeKind attribute*), 725
- `__module__` (*rti.connexdds.TypeKind.TypeKind attribute*), 724
- `__module__` (*rti.connexdds.TypeSupport attribute*), 726
- `__module__` (*rti.connexdds.Uint8Seq attribute*), 735
- `__module__` (*rti.connexdds.Uint8Type attribute*), 736
- `__module__` (*rti.connexdds.Uint16Seq attribute*), 727
- `__module__` (*rti.connexdds.Uint16Type attribute*), 729
- `__module__` (*rti.connexdds.Uint32Seq attribute*), 730
- `__module__` (*rti.connexdds.Uint32Type attribute*), 731
- `__module__` (*rti.connexdds.Uint64Seq attribute*), 732
- `__module__` (*rti.connexdds.Uint64Type attribute*), 734
- `__module__` (*rti.connexdds.UnidimensionalCollectionType attribute*), 737
- `__module__` (*rti.connexdds.UnionMember attribute*), 737
- `__module__` (*rti.connexdds.UnionMemberSeq attribute*), 739
- `__module__` (*rti.connexdds.UnionType attribute*), 740
- `__module__` (*rti.connexdds.UnsupportedError attribute*), 741
- `__module__` (*rti.connexdds.UserData attribute*), 741
- `__module__` (*rti.connexdds.UserDataSample attribute*), 741
- `__module__` (*rti.connexdds.VendorId attribute*), 742
- `__module__` (*rti.connexdds.Verbosity attribute*), 744

- `__module__` (*rti.connexdds.Verbosity.Verbosity* attribute), 743
- `__module__` (*rti.connexdds.ViewState* attribute), 745
- `__module__` (*rti.connexdds.WaitSet* attribute), 747
- `__module__` (*rti.connexdds.WaitSetProperty* attribute), 749
- `__module__` (*rti.connexdds.WcharSeq* attribute), 750
- `__module__` (*rti.connexdds.WcharType* attribute), 752
- `__module__` (*rti.connexdds.WireProtocol* attribute), 752
- `__module__` (*rti.connexdds.WireProtocolAutoKind* attribute), 755
- `__module__` (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* attribute), 754
- `__module__` (*rti.connexdds.WriteParams* attribute), 755
- `__module__` (*rti.connexdds.WriterDataLifecycle* attribute), 756
- `__module__` (*rti.connexdds.WstringSeq* attribute), 757
- `__module__` (*rti.connexdds.WstringType* attribute), 747
- `__mul__` () (*rti.connexdds.AnyDataReaderSeq* method), 44
- `__mul__` () (*rti.connexdds.AnyDataWriterSeq* method), 47
- `__mul__` () (*rti.connexdds.AnyTopicSeq* method), 50
- `__mul__` () (*rti.connexdds.BoolSeq* method), 55
- `__mul__` () (*rti.connexdds.ChannelSettingsSeq* method), 64
- `__mul__` () (*rti.connexdds.CharSeq* method), 66
- `__mul__` () (*rti.connexdds.ConditionSeq* method), 74
- `__mul__` () (*rti.connexdds.ContentFilteredTopicSeq* method), 78
- `__mul__` () (*rti.connexdds.CookieSeq* method), 81
- `__mul__` () (*rti.connexdds.DataReaderSeq* method), 108
- `__mul__` () (*rti.connexdds.DataWriterSeq* method), 139
- `__mul__` () (*rti.connexdds.DomainParticipantSeq* method), 177
- `__mul__` () (*rti.connexdds.Duration* method), 183
- `__mul__` () (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 187
- `__mul__` () (*rti.connexdds.DynamicData.DataReaderSeq* method), 196
- `__mul__` () (*rti.connexdds.DynamicData.DataWriterSeq* method), 207
- `__mul__` () (*rti.connexdds.DynamicDataSeq* method), 250
- `__mul__` () (*rti.connexdds.DynamicDataTimestampedSeq* method), 253
- `__mul__` () (*rti.connexdds.DynamicData.TopicSeq* method), 217
- `__mul__` () (*rti.connexdds.EndpointGroupSeq* method), 257
- `__mul__` () (*rti.connexdds.EntitySeq* method), 262
- `__mul__` () (*rti.connexdds.EnumMemberSeq* method), 265
- `__mul__` () (*rti.connexdds.Float32Seq* method), 275
- `__mul__` () (*rti.connexdds.Float64Seq* method), 278
- `__mul__` () (*rti.connexdds.Float128Seq* method), 273
- `__mul__` () (*rti.connexdds.IAnyDataReaderSeq* method), 292
- `__mul__` () (*rti.connexdds.IAnyDataWriterSeq* method), 295
- `__mul__` () (*rti.connexdds.IAnyTopicSeq* method), 298
- `__mul__` () (*rti.connexdds.IConditionSeq* method), 300
- `__mul__` () (*rti.connexdds.IEntitySeq* method), 303
- `__mul__` () (*rti.connexdds.InstanceHandleSeq* method), 310
- `__mul__` () (*rti.connexdds.Int8Seq* method), 324
- `__mul__` () (*rti.connexdds.Int16Seq* method), 315
- `__mul__` () (*rti.connexdds.Int32Seq* method), 318
- `__mul__` () (*rti.connexdds.Int64Seq* method), 321
- `__mul__` () (*rti.connexdds.LocatorFilterElementSeq* method), 334
- `__mul__` () (*rti.connexdds.LocatorSeq* method), 339
- `__mul__` () (*rti.connexdds.MemberSeq* method), 348
- `__mul__` () (*rti.connexdds.MonitoringMetricSelectionSeq* method), 354
- `__mul__` () (*rti.connexdds.MulticastMappingSeq* method), 359
- `__mul__` () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 372
- `__mul__` () (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 382
- `__mul__` () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 393
- `__mul__` () (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 407
- `__mul__` () (*rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq* method), 410
- `__mul__` () (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 402
- `__mul__` () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 428
- `__mul__` () (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 438
- `__mul__` () (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 450
- `__mul__` () (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 465
- `__mul__` () (*rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq* method), 467
- `__mul__` () (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 459
- `__mul__` () (*rti.connexdds.PublisherSeq* method), 477
- `__mul__` () (*rti.connexdds.QosPolicyCountSeq* method), 479
- `__mul__` () (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq* method), 526
- `__mul__` () (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 535
- `__mul__` () (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 546
- `__mul__` () (*rti.connexdds.ServiceRequestSeq* method), 561
- `__mul__` () (*rti.connexdds.ServiceRequestTimestampedSeq* method), 564
- `__mul__` () (*rti.connexdds.ServiceRequest.TopicSeq* method), 554
- `__mul__` () (*rti.connexdds.StringPairSeq* method), 573
- `__mul__` () (*rti.connexdds.StringSeq* method), 575
- `__mul__` () (*rti.connexdds.SubscriberSeq* method), 583

- `__mul__` () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 587
- `__mul__` () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 597
- `__mul__` () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 608
- `__mul__` () (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 623
- `__mul__` () (*rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq* method), 626
- `__mul__` () (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 617
- `__mul__` () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 642
- `__mul__` () (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq* method), 652
- `__mul__` () (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq* method), 663
- `__mul__` () (*rti.connexdds.TopicBuiltinTopicDataSeq* method), 677
- `__mul__` () (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq* method), 679
- `__mul__` () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq* method), 672
- `__mul__` () (*rti.connexdds.TopicSeq* method), 691
- `__mul__` () (*rti.connexdds.TransportInfoSeq* method), 700
- `__mul__` () (*rti.connexdds.TransportMulticastSeq* method), 707
- `__mul__` () (*rti.connexdds.TransportMulticastSettingsSeq* method), 710
- `__mul__` () (*rti.connexdds.TransportUnicastSettingsSeq* method), 714
- `__mul__` () (*rti.connexdds.Uint8Seq* method), 735
- `__mul__` () (*rti.connexdds.Uint16Seq* method), 727
- `__mul__` () (*rti.connexdds.Uint32Seq* method), 730
- `__mul__` () (*rti.connexdds.Uint64Seq* method), 732
- `__mul__` () (*rti.connexdds.UnionMemberSeq* method), 739
- `__mul__` () (*rti.connexdds.WcharSeq* method), 750
- `__mul__` () (*rti.connexdds.WstringSeq* method), 757
- `__ne__` () (*rti.connexdds.AcknowledgmentKind* method), 37
- `__ne__` () (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind* method), 36
- `__ne__` () (*rti.connexdds.ACTEnumMember* method), 31
- `__ne__` () (*rti.connexdds.ActivityContextMask* method), 39
- `__ne__` () (*rti.connexdds.ACTMember* method), 32
- `__ne__` () (*rti.connexdds.ACTUnionMember* method), 33
- `__ne__` () (*rti.connexdds.AliasType* method), 41
- `__ne__` () (*rti.connexdds.AllocationSettings* method), 41
- `__ne__` () (*rti.connexdds.AnyDataReaderSeq* method), 44
- `__ne__` () (*rti.connexdds.AnyDataWriterSeq* method), 47
- `__ne__` () (*rti.connexdds.AnyTopicSeq* method), 50
- `__ne__` () (*rti.connexdds.ArrayType* method), 51
- `__ne__` () (*rti.connexdds.AsynchronousPublisher* method), 51
- `__ne__` () (*rti.connexdds.Availability* method), 52
- `__ne__` () (*rti.connexdds.Batch* method), 53
- `__ne__` () (*rti.connexdds.BoolSeq* method), 55
- `__ne__` () (*rti.connexdds.BoolType* method), 56
- `__ne__` () (*rti.connexdds.BuiltinTopicKey* method), 58
- `__ne__` () (*rti.connexdds.BuiltinTopicReaderResourceLimits* method), 59
- `__ne__` () (*rti.connexdds.ByteVector* method), 61
- `__ne__` () (*rti.connexdds.CdrPaddingKind* method), 63
- `__ne__` () (*rti.connexdds.CdrPaddingKind.CdrPaddingKind* method), 62
- `__ne__` () (*rti.connexdds.ChannelSettings* method), 63
- `__ne__` () (*rti.connexdds.ChannelSettingsSeq* method), 64
- `__ne__` () (*rti.connexdds.CharSeq* method), 67
- `__ne__` () (*rti.connexdds.CoherentSetInfo* method), 69
- `__ne__` () (*rti.connexdds.CollectionType* method), 69
- `__ne__` () (*rti.connexdds.CompressionIdMask* method), 71
- `__ne__` () (*rti.connexdds.CompressionSettings* method), 73
- `__ne__` () (*rti.connexdds.ConditionSeq* method), 74
- `__ne__` () (*rti.connexdds.ContentFilteredTopic* method), 77
- `__ne__` () (*rti.connexdds.ContentFilteredTopicSeq* method), 78
- `__ne__` () (*rti.connexdds.ContentFilterProperty* method), 76
- `__ne__` () (*rti.connexdds.Cookie* method), 80
- `__ne__` () (*rti.connexdds.CookieSeq* method), 81
- `__ne__` () (*rti.connexdds.CookieVector* method), 83
- `__ne__` () (*rti.connexdds.Database* method), 141
- `__ne__` () (*rti.connexdds.DataReader* method), 86
- `__ne__` () (*rti.connexdds.DataReaderInstanceRemovalKind* method), 94
- `__ne__` () (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind* method), 92
- `__ne__` () (*rti.connexdds.DataReaderProtocol* method), 95
- `__ne__` () (*rti.connexdds.DataReaderQos* method), 100
- `__ne__` () (*rti.connexdds.DataReaderResourceLimits* method), 104
- `__ne__` () (*rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings* method), 107
- `__ne__` () (*rti.connexdds.DataReaderSeq* method), 108
- `__ne__` () (*rti.connexdds.DataRepresentation* method), 109
- `__ne__` () (*rti.connexdds.DataState* method), 111
- `__ne__` () (*rti.connexdds.DataStateEx* method), 112
- `__ne__` () (*rti.connexdds.DataTag* method), 113
- `__ne__` () (*rti.connexdds.DataWriter* method), 115
- `__ne__` () (*rti.connexdds.DataWriterProtocol* method), 122
- `__ne__` () (*rti.connexdds.DataWriterQos* method), 127
- `__ne__` () (*rti.connexdds.DataWriterResourceLimits* method), 133
- `__ne__` () (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind* method), 137
- `__ne__` () (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind* method),

- 136
- `__ne__` () (*rti.connexdds.DataWriterSeq* method), 139
- `__ne__` () (*rti.connexdds.DataWriterShmemRefTransferModeSettings* method), 140
- `__ne__` () (*rti.connexdds.DataWriterTransferMode* method), 140
- `__ne__` () (*rti.connexdds.Deadline* method), 141
- `__ne__` () (*rti.connexdds.DestinationOrder* method), 142
- `__ne__` () (*rti.connexdds.DestinationOrderKind* method), 144
- `__ne__` () (*rti.connexdds.DestinationOrderKind.DestinationOrderKind* method), 143
- `__ne__` () (*rti.connexdds.DestinationOrderScopeKind* method), 146
- `__ne__` () (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind* method), 145
- `__ne__` () (*rti.connexdds.Discovery* method), 147
- `__ne__` () (*rti.connexdds.DiscoveryConfig* method), 148
- `__ne__` () (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 152
- `__ne__` () (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 155
- `__ne__` () (*rti.connexdds.DomainParticipant* method), 158
- `__ne__` () (*rti.connexdds.DomainParticipantConfigParams* method), 164
- `__ne__` () (*rti.connexdds.DomainParticipantFactoryQos* method), 165
- `__ne__` () (*rti.connexdds.DomainParticipantQos* method), 168
- `__ne__` () (*rti.connexdds.DomainParticipantResourceLimits* method), 171
- `__ne__` () (*rti.connexdds.DomainParticipantSeq* method), 177
- `__ne__` () (*rti.connexdds.Durability* method), 178
- `__ne__` () (*rti.connexdds.DurabilityKind* method), 181
- `__ne__` () (*rti.connexdds.DurabilityKind.DurabilityKind* method), 180
- `__ne__` () (*rti.connexdds.DurabilityService* method), 182
- `__ne__` () (*rti.connexdds.Duration* method), 183
- `__ne__` () (*rti.connexdds.DynamicData* method), 221
- `__ne__` () (*rti.connexdds.DynamicData.ContentFilteredTopic* method), 185
- `__ne__` () (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 187
- `__ne__` () (*rti.connexdds.DynamicData.DataReader* method), 190
- `__ne__` () (*rti.connexdds.DynamicData.DataReaderSeq* method), 196
- `__ne__` () (*rti.connexdds.DynamicData.DataWriter* method), 198
- `__ne__` () (*rti.connexdds.DynamicData.DataWriterSeq* method), 207
- `__ne__` () (*rti.connexdds.DynamicDataEncapsulationKind* method), 247
- `__ne__` () (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind* method), 246
- `__ne__` () (*rti.connexdds.DynamicDataInfo* method), 248
- `__ne__` () (*rti.connexdds.DynamicDataMemberInfo* method), 248
- `__ne__` () (*rti.connexdds.DynamicDataProperty* method), 249
- `__ne__` () (*rti.connexdds.DynamicDataSeq* method), 250
- `__ne__` () (*rti.connexdds.DynamicDataTimestampedSeq* method), 253
- `__ne__` () (*rti.connexdds.DynamicData.Topic* method), 215
- `__ne__` () (*rti.connexdds.DynamicData.TopicDescription* method), 216
- `__ne__` () (*rti.connexdds.DynamicData.TopicSeq* method), 217
- `__ne__` () (*rti.connexdds.DynamicDataTypeSerializationProperty* method), 254
- `__ne__` () (*rti.connexdds.DynamicType* method), 255
- `__ne__` () (*rti.connexdds.DynamicTypePrintFormatProperty* method), 256
- `__ne__` () (*rti.connexdds.EndpointGroupSeq* method), 257
- `__ne__` () (*rti.connexdds.EndpointGroupVector* method), 259
- `__ne__` () (*rti.connexdds.EntityFactory* method), 260
- `__ne__` () (*rti.connexdds.EntityName* method), 261
- `__ne__` () (*rti.connexdds.EntitySeq* method), 262
- `__ne__` () (*rti.connexdds.EnumMember* method), 263
- `__ne__` () (*rti.connexdds.EnumMemberSeq* method), 265
- `__ne__` () (*rti.connexdds.EnumType* method), 266
- `__ne__` () (*rti.connexdds.Event* method), 267
- `__ne__` () (*rti.connexdds.ExclusiveArea* method), 268
- `__ne__` () (*rti.connexdds.ExpressionProperty* method), 269
- `__ne__` () (*rti.connexdds.ExtensibilityKind* method), 271
- `__ne__` () (*rti.connexdds.ExtensibilityKind.ExtensibilityKind* method), 270
- `__ne__` () (*rti.connexdds.FilterSampleInfo* method), 272
- `__ne__` () (*rti.connexdds.Float32Seq* method), 275
- `__ne__` () (*rti.connexdds.Float32Type* method), 277
- `__ne__` () (*rti.connexdds.Float64Seq* method), 278
- `__ne__` () (*rti.connexdds.Float64Type* method), 279
- `__ne__` () (*rti.connexdds.Float128Seq* method), 273
- `__ne__` () (*rti.connexdds.Float128Type* method), 274
- `__ne__` () (*rti.connexdds.FlowController* method), 280
- `__ne__` () (*rti.connexdds.FlowControllerProperty* method), 281
- `__ne__` () (*rti.connexdds.FlowControllerSchedulingPolicy* method), 284
- `__ne__` () (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy* method), 283
- `__ne__` () (*rti.connexdds.FlowControllerTokenBucketProperty* method), 285
- `__ne__` () (*rti.connexdds.GroupData* method), 286
- `__ne__` () (*rti.connexdds.GuardCondition* method), 286
- `__ne__` () (*rti.connexdds.Guid* method), 287
- `__ne__` () (*rti.connexdds.History* method), 288
- `__ne__` () (*rti.connexdds.HistoryKind* method), 290
- `__ne__` () (*rti.connexdds.HistoryKind.HistoryKind* method), 289
- `__ne__` () (*rti.connexdds.IAnyDataReader* method), 291
- `__ne__` () (*rti.connexdds.IAnyDataReaderSeq* method), 292

- `__ne__()` (*rti.connexdds.IAnyDataWriter* method), 294
- `__ne__()` (*rti.connexdds.IAnyDataWriterSeq* method), 295
- `__ne__()` (*rti.connexdds.IAnyTopic* method), 296
- `__ne__()` (*rti.connexdds.IAnyTopicSeq* method), 298
- `__ne__()` (*rti.connexdds.ICondition* method), 299
- `__ne__()` (*rti.connexdds.IConditionSeq* method), 300
- `__ne__()` (*rti.connexdds.IEntity* method), 302
- `__ne__()` (*rti.connexdds.IEntitySeq* method), 303
- `__ne__()` (*rti.connexdds.IgnoredEntityReplacementKind* method), 307
- `__ne__()` (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind* method), 306
- `__ne__()` (*rti.connexdds.InstanceHandle* method), 308
- `__ne__()` (*rti.connexdds.InstanceHandleSeq* method), 310
- `__ne__()` (*rti.connexdds.InstanceState* method), 312
- `__ne__()` (*rti.connexdds.InstanceStateConsistencyKind* method), 314
- `__ne__()` (*rti.connexdds.Int8Seq* method), 324
- `__ne__()` (*rti.connexdds.Int8Type* method), 325
- `__ne__()` (*rti.connexdds.Int16Seq* method), 315
- `__ne__()` (*rti.connexdds.Int16Type* method), 317
- `__ne__()` (*rti.connexdds.Int32Seq* method), 318
- `__ne__()` (*rti.connexdds.Int32Type* method), 319
- `__ne__()` (*rti.connexdds.Int32Vector* method), 320
- `__ne__()` (*rti.connexdds.Int64Seq* method), 322
- `__ne__()` (*rti.connexdds.Int64Type* method), 323
- `__ne__()` (*rti.connexdds.IReadCondition* method), 305
- `__ne__()` (*rti.connexdds.LatencyBudget* method), 326
- `__ne__()` (*rti.connexdds.Lifespan* method), 327
- `__ne__()` (*rti.connexdds.Liveliness* method), 327
- `__ne__()` (*rti.connexdds.LivelinessKind* method), 330
- `__ne__()` (*rti.connexdds.LivelinessKind.LivelinessKind* method), 329
- `__ne__()` (*rti.connexdds.Locator* method), 331
- `__ne__()` (*rti.connexdds.LocatorFilter* method), 332
- `__ne__()` (*rti.connexdds.LocatorFilterElement* method), 332
- `__ne__()` (*rti.connexdds.LocatorFilterElementSeq* method), 334
- `__ne__()` (*rti.connexdds.LocatorKind* method), 338
- `__ne__()` (*rti.connexdds.LocatorKind.LocatorKind* method), 337
- `__ne__()` (*rti.connexdds.LocatorSeq* method), 339
- `__ne__()` (*rti.connexdds.LocatorVector* method), 341
- `__ne__()` (*rti.connexdds.LogCategory* method), 343
- `__ne__()` (*rti.connexdds.LogCategory.LogCategory* method), 342
- `__ne__()` (*rti.connexdds.LongDouble* method), 345
- `__ne__()` (*rti.connexdds.Member* method), 346
- `__ne__()` (*rti.connexdds.MemberSeq* method), 348
- `__ne__()` (*rti.connexdds.Monitoring* method), 349
- `__ne__()` (*rti.connexdds.MonitoringDedicatedParticipantSettings* method), 350
- `__ne__()` (*rti.connexdds.MonitoringDistributionSettings* method), 350
- `__ne__()` (*rti.connexdds.MonitoringEventDistributionSettings* method), 351
- `__ne__()` (*rti.connexdds.MonitoringLoggingDistributionSettings* method), 352
- `__ne__()` (*rti.connexdds.MonitoringLoggingForwardingSettings* method), 352
- `__ne__()` (*rti.connexdds.MonitoringMetricSelection* method), 353
- `__ne__()` (*rti.connexdds.MonitoringMetricSelectionSeq* method), 354
- `__ne__()` (*rti.connexdds.MonitoringPeriodicDistributionSettings* method), 356
- `__ne__()` (*rti.connexdds.MonitoringTelemetryData* method), 356
- `__ne__()` (*rti.connexdds.MulticastMapping* method), 358
- `__ne__()` (*rti.connexdds.MulticastMappingSeq* method), 359
- `__ne__()` (*rti.connexdds.MultiChannel* method), 357
- `__ne__()` (*rti.connexdds.Ownership* method), 367
- `__ne__()` (*rti.connexdds.OwnershipKind* method), 369
- `__ne__()` (*rti.connexdds.OwnershipKind.OwnershipKind* method), 368
- `__ne__()` (*rti.connexdds.OwnershipStrength* method), 369
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData* method), 405
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic* method), 370
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 372
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* method), 376
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 382
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter* method), 384
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 393
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 407
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq* method), 410
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData.Topic* method), 400
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicDescription* method), 401
- `__ne__()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 402
- `__ne__()` (*rti.connexdds.Partition* method), 411
- `__ne__()` (*rti.connexdds.PersistentJournalKind* method), 412
- `__ne__()` (*rti.connexdds.PersistentStorageSettings* method), 413

- `__ne__()` (*rti.connexdds.PersistentSynchronizationKind* method), 414
- `__ne__()` (*rti.connexdds.Presentation* method), 415
- `__ne__()` (*rti.connexdds.PresentationAccessScopeKind* method), 418
- `__ne__()` (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind* method), 417
- `__ne__()` (*rti.connexdds.PrintFormat* method), 420
- `__ne__()` (*rti.connexdds.PrintFormatKind* method), 422
- `__ne__()` (*rti.connexdds.PrintFormatKind.PrintFormatKind* method), 421
- `__ne__()` (*rti.connexdds.PrintFormat.PrintFormat* method), 419
- `__ne__()` (*rti.connexdds.ProductVersion* method), 423
- `__ne__()` (*rti.connexdds.Property* method), 424
- `__ne__()` (*rti.connexdds.ProtocolVersion* method), 425
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicData* method), 462
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic* method), 427
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 428
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader* method), 432
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 438
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter* method), 441
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 450
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 465
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq* method), 467
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicData.Topic* method), 456
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicDescription* method), 457
- `__ne__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 459
- `__ne__()` (*rti.connexdds.Publisher* method), 472
- `__ne__()` (*rti.connexdds.PublisherQos* method), 475
- `__ne__()` (*rti.connexdds.PublisherSeq* method), 477
- `__ne__()` (*rti.connexdds.PublishMode* method), 469
- `__ne__()` (*rti.connexdds.PublishModeKind* method), 472
- `__ne__()` (*rti.connexdds.PublishModeKind.PublishModeKind* method), 471
- `__ne__()` (*rti.connexdds.QosPolicyCount* method), 478
- `__ne__()` (*rti.connexdds.QosPolicyCountSeq* method), 479
- `__ne__()` (*rti.connexdds.QosPrintFormat* method), 481
- `__ne__()` (*rti.connexdds.QosProvider* method), 481
- `__ne__()` (*rti.connexdds.QosProviderParams* method), 484
- `__ne__()` (*rti.connexdds.Query* method), 485
- `__ne__()` (*rti.connexdds.QueryCondition* method), 486
- `__ne__()` (*rti.connexdds.ReaderDataLifecycle* method), 488
- `__ne__()` (*rti.connexdds.ReceiverPool* method), 489
- `__ne__()` (*rti.connexdds.Reliability* method), 490
- `__ne__()` (*rti.connexdds.ReliabilityKind* method), 492
- `__ne__()` (*rti.connexdds.ReliabilityKind.ReliabilityKind* method), 491
- `__ne__()` (*rti.connexdds.RemoteParticipantPurgeKind* method), 495
- `__ne__()` (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* method), 494
- `__ne__()` (*rti.connexdds.ResourceLimits* method), 497
- `__ne__()` (*rti.connexdds.RtpsReliableReaderProtocol* method), 497
- `__ne__()` (*rti.connexdds.RtpsReliableWriterProtocol* method), 498
- `__ne__()` (*rti.connexdds.RtpsReservedPortKindMask* method), 502
- `__ne__()` (*rti.connexdds.RtpsWellKnownPorts* method), 504
- `__ne__()` (*rti.connexdds.SampleFlag* method), 506
- `__ne__()` (*rti.connexdds.SampleIdentity* method), 507
- `__ne__()` (*rti.connexdds.SampleInfo* method), 508
- `__ne__()` (*rti.connexdds.SampleLostState* method), 511
- `__ne__()` (*rti.connexdds.SampleRejectedState* method), 514
- `__ne__()` (*rti.connexdds.SampleState* method), 517
- `__ne__()` (*rti.connexdds.SequenceNumber* method), 520
- `__ne__()` (*rti.connexdds.SequenceType* method), 520
- `__ne__()` (*rti.connexdds.Service* method), 521
- `__ne__()` (*rti.connexdds.ServiceKind* method), 523
- `__ne__()` (*rti.connexdds.ServiceKind.ServiceKind* method), 522
- `__ne__()` (*rti.connexdds.ServiceRequest* method), 557
- `__ne__()` (*rti.connexdds.ServiceRequest.ContentFilteredTopic* method), 524
- `__ne__()` (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq* method), 526
- `__ne__()` (*rti.connexdds.ServiceRequest.DataReader* method), 530
- `__ne__()` (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 535
- `__ne__()` (*rti.connexdds.ServiceRequest.DataWriter* method), 537
- `__ne__()` (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 546
- `__ne__()` (*rti.connexdds.ServiceRequestId* method), 560
- `__ne__()` (*rti.connexdds.ServiceRequestId.ServiceRequestId* method), 559
- `__ne__()` (*rti.connexdds.ServiceRequestSeq* method), 561
- `__ne__()` (*rti.connexdds.ServiceRequestTimestampedSeq* method), 564
- `__ne__()` (*rti.connexdds.ServiceRequest.Topic* method), 552
- `__ne__()` (*rti.connexdds.ServiceRequest.TopicDescription* method), 553

- `__ne__()` (*rti.connexdds.ServiceRequest.TopicSeq* method), 555
- `__ne__()` (*rti.connexdds.StatusCondition* method), 565
- `__ne__()` (*rti.connexdds.StatusMask* method), 568
- `__ne__()` (*rti.connexdds.StreamKind* method), 570
- `__ne__()` (*rti.connexdds.StringPairSeq* method), 573
- `__ne__()` (*rti.connexdds.StringSeq* method), 575
- `__ne__()` (*rti.connexdds.StringType* method), 577
- `__ne__()` (*rti.connexdds.StructType* method), 578
- `__ne__()` (*rti.connexdds.Subscriber* method), 579
- `__ne__()` (*rti.connexdds.SubscriberQos* method), 581
- `__ne__()` (*rti.connexdds.SubscriberSeq* method), 583
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicData* method), 620
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic* method), 585
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 587
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader* method), 591
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 597
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* method), 599
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 608
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 623
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq* method), 626
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicData.Topic* method), 615
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicDescription* method), 616
- `__ne__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 617
- `__ne__()` (*rti.connexdds.SyslogVerbosity* method), 629
- `__ne__()` (*rti.connexdds.SystemResourceLimits* method), 630
- `__ne__()` (*rti.connexdds.ThreadSettings* method), 631
- `__ne__()` (*rti.connexdds.ThreadSettingsCpuRotationKind* method), 633
- `__ne__()` (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* method), 632
- `__ne__()` (*rti.connexdds.ThreadSettingsKindMask* method), 634
- `__ne__()` (*rti.connexdds.Time* method), 637
- `__ne__()` (*rti.connexdds.TimeBasedFilter* method), 638
- `__ne__()` (*rti.connexdds.Topic* method), 639
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicData* method), 675
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic* method), 640
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 642
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicData.DataReader* method), 646
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq* method), 652
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriter* method), 654
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq* method), 663
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicDataSeq* method), 677
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq* method), 679
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicData.Topic* method), 669
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicData.TopicDescription* method), 670
- `__ne__()` (*rti.connexdds.TopicBuiltinTopicData.TopicSeq* method), 672
- `__ne__()` (*rti.connexdds.TopicData* method), 681
- `__ne__()` (*rti.connexdds.TopicDescription* method), 681
- `__ne__()` (*rti.connexdds.TopicQos* method), 683
- `__ne__()` (*rti.connexdds.TopicQuery* method), 686
- `__ne__()` (*rti.connexdds.TopicQueryDispatch* method), 687
- `__ne__()` (*rti.connexdds.TopicQuerySelectionKind* method), 690
- `__ne__()` (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind* method), 689
- `__ne__()` (*rti.connexdds.TopicSeq* method), 691
- `__ne__()` (*rti.connexdds.TransportBuiltin* method), 692
- `__ne__()` (*rti.connexdds.TransportBuiltinMask* method), 694
- `__ne__()` (*rti.connexdds.TransportClassId* method), 699
- `__ne__()` (*rti.connexdds.TransportClassId.TransportClassId* method), 698
- `__ne__()` (*rti.connexdds.TransportInfoSeq* method), 700
- `__ne__()` (*rti.connexdds.TransportInfoVector* method), 702
- `__ne__()` (*rti.connexdds.TransportMulticast* method), 703
- `__ne__()` (*rti.connexdds.TransportMulticastKind* method), 705
- `__ne__()` (*rti.connexdds.TransportMulticastKind.TransportMulticastKind* method), 703
- `__ne__()` (*rti.connexdds.TransportMulticastMapping* method), 705
- `__ne__()` (*rti.connexdds.TransportMulticastMappingFunction* method), 706
- `__ne__()` (*rti.connexdds.TransportMulticastSeq* method), 707
- `__ne__()` (*rti.connexdds.TransportMulticastSettings* method), 708

- `__ne__()` (*rti.connexdds.TransportMulticastSettingsSeq method*), 710
- `__ne__()` (*rti.connexdds.TransportPriority method*), 711
- `__ne__()` (*rti.connexdds.TransportSelection method*), 712
- `__ne__()` (*rti.connexdds.TransportUnicast method*), 712
- `__ne__()` (*rti.connexdds.TransportUnicastSettings method*), 713
- `__ne__()` (*rti.connexdds.TransportUnicastSettingsSeq method*), 714
- `__ne__()` (*rti.connexdds.TypeConsistencyEnforcement method*), 717
- `__ne__()` (*rti.connexdds.TypeConsistencyEnforcementKind method*), 719
- `__ne__()` (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 718
- `__ne__()` (*rti.connexdds.TypeKind method*), 725
- `__ne__()` (*rti.connexdds.TypeKind.TypeKind method*), 724
- `__ne__()` (*rti.connexdds.TypeSupport method*), 726
- `__ne__()` (*rti.connexdds.Uint8Seq method*), 735
- `__ne__()` (*rti.connexdds.Uint8Type method*), 736
- `__ne__()` (*rti.connexdds.Uint16Seq method*), 727
- `__ne__()` (*rti.connexdds.Uint16Type method*), 729
- `__ne__()` (*rti.connexdds.Uint32Seq method*), 730
- `__ne__()` (*rti.connexdds.Uint32Type method*), 731
- `__ne__()` (*rti.connexdds.Uint64Seq method*), 732
- `__ne__()` (*rti.connexdds.Uint64Type method*), 734
- `__ne__()` (*rti.connexdds.UnidimensionalCollectionType method*), 737
- `__ne__()` (*rti.connexdds.UnionMember method*), 737
- `__ne__()` (*rti.connexdds.UnionMemberSeq method*), 739
- `__ne__()` (*rti.connexdds.UnionType method*), 740
- `__ne__()` (*rti.connexdds.UserData method*), 741
- `__ne__()` (*rti.connexdds.VendorId method*), 742
- `__ne__()` (*rti.connexdds.Verbosity method*), 744
- `__ne__()` (*rti.connexdds.Verbosity.Verbosity method*), 743
- `__ne__()` (*rti.connexdds.ViewState method*), 745
- `__ne__()` (*rti.connexdds.WaitSetProperty method*), 749
- `__ne__()` (*rti.connexdds.WcharSeq method*), 750
- `__ne__()` (*rti.connexdds.WcharType method*), 752
- `__ne__()` (*rti.connexdds.WireProtocol method*), 752
- `__ne__()` (*rti.connexdds.WireProtocolAutoKind method*), 755
- `__ne__()` (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 754
- `__ne__()` (*rti.connexdds.WriterDataLifecycle method*), 756
- `__ne__()` (*rti.connexdds.WstringSeq method*), 757
- `__ne__()` (*rti.connexdds.WstringType method*), 747
- `__next__()` (*rti.connexdds.DynamicData.FieldsIterator method*), 208
- `__next__()` (*rti.connexdds.DynamicData.IndexIterator method*), 209
- `__next__()` (*rti.connexdds.DynamicData.ItemsIterator method*), 209
- `__next__()` (*rti.connexdds.TriggeredConditionsIterator method*), 716
- `__nonzero__()` (*rti.connexdds.InstanceHandle method*), 308
- `__or__()` (*rti.connexdds.ActivityContextMask method*), 39
- `__or__()` (*rti.connexdds.CompressionIdMask method*), 71
- `__or__()` (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method*), 152
- `__or__()` (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method*), 155
- `__or__()` (*rti.connexdds.InstanceState method*), 312
- `__or__()` (*rti.connexdds.RtpsReservedPortKindMask method*), 502
- `__or__()` (*rti.connexdds.SampleFlag method*), 506
- `__or__()` (*rti.connexdds.SampleLostState method*), 511
- `__or__()` (*rti.connexdds.SampleRejectedState method*), 515
- `__or__()` (*rti.connexdds.SampleState method*), 517
- `__or__()` (*rti.connexdds.StatusMask method*), 568
- `__or__()` (*rti.connexdds.StreamKind method*), 570
- `__or__()` (*rti.connexdds.ThreadSettingsKindMask method*), 634
- `__or__()` (*rti.connexdds.TransportBuiltinMask method*), 694
- `__or__()` (*rti.connexdds.ViewState method*), 746
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.BoolSeq attribute*), 55
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.CharSeq attribute*), 67
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.DynamicDataTimestampedSeq attribute*), 253
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.Float32Seq attribute*), 275
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.Float64Seq attribute*), 278
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.IAnyDataReaderSeq attribute*), 292
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.IAnyDataWriterSeq attribute*), 295
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.IAnyTopicSeq attribute*), 298
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.IConditionSeq attribute*), 300
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.IEntitySeq attribute*), 303
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.Int8Seq attribute*), 324
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.Int16Seq attribute*), 315
- `__pybind11_module_local_v4_gcc_libstdcpp_cxxabi1002__` (*rti.connexdds.Int32Seq attribute*), 318

`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__` (*rti.connexdds.Int64Seq
 attribute*), 322
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__`
 (*rti.connexdds.ParticipantBuiltinTopicDataTimes-
 tampedSeq attribute*),
 410
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__`
 (*rti.connexdds.PublicationBuiltinTopicDataTimes-
 tampedSeq attribute*),
 467
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__`
 (*rti.connexdds.ServiceRequestTimestampedSeq
 attribute*), 564
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__` (*rti.connexdds.StringMap
 attribute*), 572
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__` (*rti.connexdds.StringPairSeq
 attribute*), 573
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__` (*rti.connexdds.StringSeq
 attribute*), 576
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__`
 (*rti.connexdds.SubscriptionBuiltinTopicDataTimes-
 tampedSeq attribute*),
 626
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__`
 (*rti.connexdds.TopicBuiltinTopicDataTimestamped-
 Seq attribute*),
 679
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__` (*rti.connexdds.Uint8Seq
 attribute*), 735
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__` (*rti.connexdds.Uint16Seq
 attribute*), 727
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__` (*rti.connexdds.Uint32Seq
 attribute*), 730
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__` (*rti.connexdds.Uint64Seq
 attribute*), 733
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__` (*rti.connexdds.WcharSeq
 attribute*), 750
`__pybind11_module_local_v4_gcc_libstd-
 cpp_cxxabi1002__` (*rti.connexdds.WstringSeq
 attribute*), 758
`__radd__` (*rti.connexdds.Time method*), 637
`__repr__` (*rti.connexdds.AcknowledgmentKind.Acknowl-
 edgmentKind method*),
 36
`__repr__` (*rti.connexdds.ActivityContextMask method*), 39
`__repr__` (*rti.connexdds.BoolSeq method*), 55
`__repr__` (*rti.connexdds.BuiltinTopicKey method*), 58
`__repr__` (*rti.connexdds.CdrPaddingKind.CdrPaddingKind
 method*), 62
`__repr__` (*rti.connexdds.CharSeq method*), 67
`__repr__` (*rti.connexdds.CompressionIdMask method*), 71
`__repr__` (*rti.connexdds.CookieSeq method*), 81
`__repr__` (*rti.connexdds.DataReaderInstanceRe-
 movalKind.DataReaderInstanceRemovalKind
 method*), 92
`__repr__` (*rti.connexdds.DataState method*), 111
`__repr__` (*rti.connexdds.DataWriterResourceLimitsIn-
 stanceReplacementKind.DataWriterResourceLimitsIn-
 stanceReplacementKind method*),
 136
`__repr__` (*rti.connexdds.DestinationOrderKind.Destina-
 tionOrderKind method*),
 143
`__repr__` (*rti.connexdds.DestinationOrderScopeKind.Desti-
 nationOrderScopeKind method*),
 145
`__repr__` (*rti.connexdds.DiscoveryConfigBuiltinChan-
 nelKindMask method*),
 152
`__repr__` (*rti.connexdds.DiscoveryConfigBuiltinPlug-
 inKindMask method*),
 155
`__repr__` (*rti.connexdds.DurabilityKind.DurabilityKind
 method*), 180
`__repr__` (*rti.connexdds.DynamicDataEncapsula-
 tionKind.DynamicDataEncapsulationKind method*),
 246
`__repr__` (*rti.connexdds.DynamicDataSeq method*), 250
`__repr__` (*rti.connexdds.ExtensibilityKind.ExtensibilityKind
 method*), 270
`__repr__` (*rti.connexdds.Float32Seq method*), 275
`__repr__` (*rti.connexdds.Float64Seq method*), 278
`__repr__` (*rti.connexdds.Float128Seq method*), 273
`__repr__` (*rti.connexdds.FlowControllerSchedulingPol-
 icy.FlowControllerSchedulingPolicy method*),
 283
`__repr__` (*rti.connexdds.HistoryKind.HistoryKind
 method*), 289
`__repr__` (*rti.connexdds.IAnyDataReaderSeq method*), 292
`__repr__` (*rti.connexdds.IAnyDataWriterSeq method*), 295
`__repr__` (*rti.connexdds.IAnyTopicSeq method*), 298
`__repr__` (*rti.connexdds.IConditionSeq method*), 300
`__repr__` (*rti.connexdds.IEntitySeq method*), 304
`__repr__` (*rti.connexdds.IgnoredEntityReplacementKind.Ig-
 noredEntityReplacementKind method*),
 306
`__repr__` (*rti.connexdds.InstanceHandleSeq method*), 310
`__repr__` (*rti.connexdds.InstanceStateConsistencyKind
 method*), 314
`__repr__` (*rti.connexdds.Int8Seq method*), 324
`__repr__` (*rti.connexdds.Int16Seq method*), 315
`__repr__` (*rti.connexdds.Int32Seq method*), 318
`__repr__` (*rti.connexdds.Int64Seq method*), 322

`__repr__()` (*rti.connexdds.LivelinessKind.LivelinessKind method*), 329
`__repr__()` (*rti.connexdds.LocatorKind.LocatorKind method*), 337
`__repr__()` (*rti.connexdds.LogCategory.LogCategory method*), 342
`__repr__()` (*rti.connexdds.OwnershipKind.OwnershipKind method*), 368
`__repr__()` (*rti.connexdds.PersistentJournalKind method*), 413
`__repr__()` (*rti.connexdds.PersistentSynchronizationKind method*), 414
`__repr__()` (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 417
`__repr__()` (*rti.connexdds.PrintFormatKind.PrintFormatKind method*), 421
`__repr__()` (*rti.connexdds.PrintFormat.PrintFormat method*), 419
`__repr__()` (*rti.connexdds.PublishModeKind.PublishModeKind method*), 471
`__repr__()` (*rti.connexdds.ReliabilityKind.ReliabilityKind method*), 491
`__repr__()` (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 494
`__repr__()` (*rti.connexdds.RtpsReservedPortKindMask method*), 502
`__repr__()` (*rti.connexdds.SampleInfo method*), 508
`__repr__()` (*rti.connexdds.SequenceNumber method*), 520
`__repr__()` (*rti.connexdds.ServiceKind.ServiceKind method*), 522
`__repr__()` (*rti.connexdds.ServiceRequestId.ServiceRequestId method*), 559
`__repr__()` (*rti.connexdds.StatusMask method*), 568
`__repr__()` (*rti.connexdds.StringMap method*), 572
`__repr__()` (*rti.connexdds.StringSeq method*), 576
`__repr__()` (*rti.connexdds.SyslogVerbosity method*), 629
`__repr__()` (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 632
`__repr__()` (*rti.connexdds.ThreadSettingsKindMask method*), 634
`__repr__()` (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 689
`__repr__()` (*rti.connexdds.TransportBuiltinMask method*), 694
`__repr__()` (*rti.connexdds.TransportClassId.TransportClassId method*), 698
`__repr__()` (*rti.connexdds.TransportMulticastKind.TransportMulticastKind method*), 704
`__repr__()` (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 718
`__repr__()` (*rti.connexdds.TypeKind.TypeKind method*), 724
`__repr__()` (*rti.connexdds.Uint8Seq method*), 735
`__repr__()` (*rti.connexdds.Uint16Seq method*), 727
`__repr__()` (*rti.connexdds.Uint32Seq method*), 730
`__repr__()` (*rti.connexdds.Uint64Seq method*), 733
`__repr__()` (*rti.connexdds.Verbosity.Verbosity method*), 743
`__repr__()` (*rti.connexdds.WcharSeq method*), 750
`__repr__()` (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 754
`__reversed__()` (*rti.connexdds.DynamicData method*), 221
`__reversed__()` (*rti.connexdds.DynamicData.FieldsView method*), 209
`__reversed__()` (*rti.connexdds.DynamicData.ItemsView method*), 210
`__reversed__()` (*rti.connexdds.TriggeredConditions method*), 716
`__rmul__()` (*rti.connexdds.AnyDataReaderSeq method*), 44
`__rmul__()` (*rti.connexdds.AnyDataWriterSeq method*), 47
`__rmul__()` (*rti.connexdds.AnyTopicSeq method*), 50
`__rmul__()` (*rti.connexdds.BoolSeq method*), 55
`__rmul__()` (*rti.connexdds.ChannelSettingsSeq method*), 64
`__rmul__()` (*rti.connexdds.CharSeq method*), 67
`__rmul__()` (*rti.connexdds.ConditionSeq method*), 75
`__rmul__()` (*rti.connexdds.ContentFilteredTopicSeq method*), 78
`__rmul__()` (*rti.connexdds.CookieSeq method*), 81
`__rmul__()` (*rti.connexdds.DataReaderSeq method*), 108
`__rmul__()` (*rti.connexdds.DataWriterSeq method*), 139
`__rmul__()` (*rti.connexdds.DomainParticipantSeq method*), 177
`__rmul__()` (*rti.connexdds.Duration method*), 183
`__rmul__()` (*rti.connexdds.DynamicData.ContentFilteredTopicSeq method*), 187
`__rmul__()` (*rti.connexdds.DynamicData.DataReaderSeq method*), 196
`__rmul__()` (*rti.connexdds.DynamicData.DataWriterSeq method*), 207
`__rmul__()` (*rti.connexdds.DynamicDataSeq method*), 250
`__rmul__()` (*rti.connexdds.DynamicDataTimestampedSeq method*), 253
`__rmul__()` (*rti.connexdds.DynamicData.TopicSeq method*), 217
`__rmul__()` (*rti.connexdds.EndpointGroupSeq method*), 257
`__rmul__()` (*rti.connexdds.EntitySeq method*), 262
`__rmul__()` (*rti.connexdds.EnumMemberSeq method*), 265
`__rmul__()` (*rti.connexdds.Float32Seq method*), 275
`__rmul__()` (*rti.connexdds.Float64Seq method*), 278
`__rmul__()` (*rti.connexdds.Float128Seq method*), 273
`__rmul__()` (*rti.connexdds.IAnyDataReaderSeq method*), 292
`__rmul__()` (*rti.connexdds.IAnyDataWriterSeq method*), 295
`__rmul__()` (*rti.connexdds.IAnyTopicSeq method*), 298
`__rmul__()` (*rti.connexdds.IConditionSeq method*), 300
`__rmul__()` (*rti.connexdds.IEntitySeq method*), 304
`__rmul__()` (*rti.connexdds.InstanceHandleSeq method*), 310
`__rmul__()` (*rti.connexdds.Int8Seq method*), 324
`__rmul__()` (*rti.connexdds.Int16Seq method*), 315
`__rmul__()` (*rti.connexdds.Int32Seq method*), 318

- `__rmul__` () (*rti.connexdds.Int64Seq* method), 322
- `__rmul__` () (*rti.connexdds.LocatorFilterElementSeq* method), 334
- `__rmul__` () (*rti.connexdds.LocatorSeq* method), 339
- `__rmul__` () (*rti.connexdds.MemberSeq* method), 348
- `__rmul__` () (*rti.connexdds.MonitoringMetricSelectionSeq* method), 355
- `__rmul__` () (*rti.connexdds.MulticastMappingSeq* method), 359
- `__rmul__` () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 372
- `__rmul__` () (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 382
- `__rmul__` () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 394
- `__rmul__` () (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 407
- `__rmul__` () (*rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq* method), 410
- `__rmul__` () (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 402
- `__rmul__` () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 429
- `__rmul__` () (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 438
- `__rmul__` () (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 450
- `__rmul__` () (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 465
- `__rmul__` () (*rti.connexdds.PublicationBuiltinTopicData.TimestampedSeq* method), 468
- `__rmul__` () (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 459
- `__rmul__` () (*rti.connexdds.PublisherSeq* method), 477
- `__rmul__` () (*rti.connexdds.QosPolicyCountSeq* method), 479
- `__rmul__` () (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq* method), 526
- `__rmul__` () (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 535
- `__rmul__` () (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 546
- `__rmul__` () (*rti.connexdds.ServiceRequestSeq* method), 561
- `__rmul__` () (*rti.connexdds.ServiceRequestTimestampedSeq* method), 564
- `__rmul__` () (*rti.connexdds.ServiceRequest.TopicSeq* method), 555
- `__rmul__` () (*rti.connexdds.StringPairSeq* method), 574
- `__rmul__` () (*rti.connexdds.StringSeq* method), 576
- `__rmul__` () (*rti.connexdds.SubscriberSeq* method), 583
- `__rmul__` () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 587
- `__rmul__` () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 597
- `__rmul__` () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 608
- `__rmul__` () (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 624
- `__rmul__` () (*rti.connexdds.SubscriptionBuiltinTopicData.TimestampedSeq* method), 626
- `__rmul__` () (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 617
- `__rmul__` () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 642
- `__rmul__` () (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq* method), 652
- `__rmul__` () (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq* method), 663
- `__rmul__` () (*rti.connexdds.TopicBuiltinTopicDataSeq* method), 677
- `__rmul__` () (*rti.connexdds.TopicBuiltinTopicData.TimestampedSeq* method), 679
- `__rmul__` () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq* method), 672
- `__rmul__` () (*rti.connexdds.TopicSeq* method), 691
- `__rmul__` () (*rti.connexdds.TransportInfoSeq* method), 700
- `__rmul__` () (*rti.connexdds.TransportMulticastSeq* method), 707
- `__rmul__` () (*rti.connexdds.TransportMulticastSettingsSeq* method), 710
- `__rmul__` () (*rti.connexdds.TransportUnicastSettingsSeq* method), 715
- `__rmul__` () (*rti.connexdds.Uint8Seq* method), 735
- `__rmul__` () (*rti.connexdds.Uint16Seq* method), 727
- `__rmul__` () (*rti.connexdds.Uint32Seq* method), 730
- `__rmul__` () (*rti.connexdds.Uint64Seq* method), 733
- `__rmul__` () (*rti.connexdds.UnionMemberSeq* method), 739
- `__rmul__` () (*rti.connexdds.WcharSeq* method), 750
- `__rmul__` () (*rti.connexdds.WstringSeq* method), 758
- `rshift` () (*rti.connexdds.ActivityContextMask* method), 39
- `rshift` () (*rti.connexdds.CompressionIdMask* method), 71
- `rshift` () (*rti.connexdds.DataReader* method), 86
- `rshift` () (*rti.connexdds.DataReaderQos* method), 100
- `rshift` () (*rti.connexdds.DataWriter* method), 115
- `rshift` () (*rti.connexdds.DataWriterQos* method), 128
- `rshift` () (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method),

- 152
 __rshift__ () (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method*),
 155
 __rshift__ () (*rti.connexdds.DomainParticipant method*),
 158
 __rshift__ () (*rti.connexdds.DomainParticipantFactoryQos method*), 165
 __rshift__ () (*rti.connexdds.DomainParticipantQos method*), 168
 __rshift__ () (*rti.connexdds.DynamicData.DataReader method*), 191
 __rshift__ () (*rti.connexdds.DynamicData.DataWriter method*), 198
 __rshift__ () (*rti.connexdds.InstanceState method*), 312
 __rshift__ () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*),
 376
 __rshift__ () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*),
 384
 __rshift__ () (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*),
 432
 __rshift__ () (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*),
 441
 __rshift__ () (*rti.connexdds.Publisher method*), 473
 __rshift__ () (*rti.connexdds.PublisherQos method*), 475
 __rshift__ () (*rti.connexdds.RtpsReservedPortKindMask method*), 502
 __rshift__ () (*rti.connexdds.SampleFlag method*), 506
 __rshift__ () (*rti.connexdds.SampleLostState method*), 512
 __rshift__ () (*rti.connexdds.SampleRejectedState method*),
 515
 __rshift__ () (*rti.connexdds.SampleState method*), 518
 __rshift__ () (*rti.connexdds.ServiceRequest.DataReader method*), 530
 __rshift__ () (*rti.connexdds.ServiceRequest.DataWriter method*), 537
 __rshift__ () (*rti.connexdds.StatusMask method*), 568
 __rshift__ () (*rti.connexdds.StreamKind method*), 570
 __rshift__ () (*rti.connexdds.SubscriberQos method*), 581
 __rshift__ () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*),
 591
 __rshift__ () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method*),
 599
 __rshift__ () (*rti.connexdds.ThreadSettingsKindMask method*), 635
 __rshift__ ()
 (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 646
 __rshift__ ()
 (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 654
 __rshift__ () (*rti.connexdds.TopicQos method*), 683
 __rshift__ () (*rti.connexdds.TransportBuiltinMask method*),
 694
 __rshift__ () (*rti.connexdds.ViewState method*), 746
 __setitem__ () (*rti.connexdds.ActivityContextMask method*),
 39
 __setitem__ () (*rti.connexdds.AnyDataReaderSeq method*),
 44
 __setitem__ () (*rti.connexdds.AnyDataWriterSeq method*),
 47
 __setitem__ () (*rti.connexdds.AnyTopicSeq method*), 50
 __setitem__ () (*rti.connexdds.BoolSeq method*), 55
 __setitem__ () (*rti.connexdds.ByteVector method*), 61
 __setitem__ () (*rti.connexdds.ChannelSettingsSeq method*),
 64
 __setitem__ () (*rti.connexdds.CharSeq method*), 67
 __setitem__ () (*rti.connexdds.CompressionIdMask method*),
 71
 __setitem__ () (*rti.connexdds.ConditionSeq method*), 75
 __setitem__ () (*rti.connexdds.ContentFilteredTopicSeq method*), 78
 __setitem__ () (*rti.connexdds.CookieSeq method*), 81
 __setitem__ () (*rti.connexdds.CookieVector method*), 83
 __setitem__ () (*rti.connexdds.DataReaderSeq method*), 108
 __setitem__ () (*rti.connexdds.DataTag method*), 113
 __setitem__ () (*rti.connexdds.DataWriterSeq method*), 139
 __setitem__ () (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method*),
 152
 __setitem__ () (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method*),
 155
 __setitem__ () (*rti.connexdds.DomainParticipantSeq method*), 177
 __setitem__ () (*rti.connexdds.DynamicData method*), 221
 __setitem__ ()
 (*rti.connexdds.DynamicData.ContentFilteredTopicSeq method*), 187
 __setitem__ ()
 (*rti.connexdds.DynamicData.DataReaderSeq method*), 196
 __setitem__ () (*rti.connexdds.DynamicData.DataWriterSeq method*), 207
 __setitem__ () (*rti.connexdds.DynamicDataSeq method*),
 251
 __setitem__ ()
 (*rti.connexdds.DynamicDataTimestampedSeq method*), 253
 __setitem__ () (*rti.connexdds.DynamicData.TopicSeq method*), 217
 __setitem__ () (*rti.connexdds.EndpointGroupSeq method*),
 257
 __setitem__ () (*rti.connexdds.EndpointGroupVector method*), 259
 __setitem__ () (*rti.connexdds.EntitySeq method*), 262
 __setitem__ () (*rti.connexdds.EnumMemberSeq method*),
 265
 __setitem__ () (*rti.connexdds.Float32Seq method*), 275
 __setitem__ () (*rti.connexdds.Float64Seq method*), 278
 __setitem__ () (*rti.connexdds.Float128Seq method*), 273

- `__setitem__()` (*rti.connexdds.Guid* method), 287
- `__setitem__()` (*rti.connexdds.IAnyDataReaderSeq* method), 292
- `__setitem__()` (*rti.connexdds.IAnyDataWriterSeq* method), 295
- `__setitem__()` (*rti.connexdds.IAnyTopicSeq* method), 298
- `__setitem__()` (*rti.connexdds.IConditionSeq* method), 301
- `__setitem__()` (*rti.connexdds.IEntitySeq* method), 304
- `__setitem__()` (*rti.connexdds.InstanceHandleSeq* method), 310
- `__setitem__()` (*rti.connexdds.InstanceState* method), 312
- `__setitem__()` (*rti.connexdds.Int8Seq* method), 324
- `__setitem__()` (*rti.connexdds.Int16Seq* method), 315
- `__setitem__()` (*rti.connexdds.Int32Seq* method), 318
- `__setitem__()` (*rti.connexdds.Int32Vector* method), 320
- `__setitem__()` (*rti.connexdds.Int64Seq* method), 322
- `__setitem__()` (*rti.connexdds.LocatorFilterElementSeq* method), 334
- `__setitem__()` (*rti.connexdds.LocatorSeq* method), 339
- `__setitem__()` (*rti.connexdds.LocatorVector* method), 341
- `__setitem__()` (*rti.connexdds.LongDouble* method), 345
- `__setitem__()` (*rti.connexdds.MemberSeq* method), 348
- `__setitem__()` (*rti.connexdds.MonitoringMetricSelectionSeq* method), 355
- `__setitem__()` (*rti.connexdds.MulticastMappingSeq* method), 359
- `__setitem__()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 372
- `__setitem__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 382
- `__setitem__()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 394
- `__setitem__()` (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 407
- `__setitem__()` (*rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq* method), 410
- `__setitem__()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 402
- `__setitem__()` (*rti.connexdds.Property* method), 424
- `__setitem__()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 429
- `__setitem__()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 438
- `__setitem__()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 450
- `__setitem__()` (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 465
- `__setitem__()` (*rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq* method), 468
- `__setitem__()` (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 459
- `__setitem__()` (*rti.connexdds.PublisherSeq* method), 477
- `__setitem__()` (*rti.connexdds.QosPolicyCountSeq* method), 479
- `__setitem__()` (*rti.connexdds.RtpsReservedPortKindMask* method), 502
- `__setitem__()` (*rti.connexdds.SampleFlag* method), 506
- `__setitem__()` (*rti.connexdds.SampleLostState* method), 512
- `__setitem__()` (*rti.connexdds.SampleRejectedState* method), 515
- `__setitem__()` (*rti.connexdds.SampleState* method), 518
- `__setitem__()` (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq* method), 526
- `__setitem__()` (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 535
- `__setitem__()` (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 546
- `__setitem__()` (*rti.connexdds.ServiceRequestSeq* method), 561
- `__setitem__()` (*rti.connexdds.ServiceRequestTimestampedSeq* method), 564
- `__setitem__()` (*rti.connexdds.ServiceRequest.TopicSeq* method), 555
- `__setitem__()` (*rti.connexdds.StatusMask* method), 568
- `__setitem__()` (*rti.connexdds.StreamKind* method), 571
- `__setitem__()` (*rti.connexdds.StringMap* method), 572
- `__setitem__()` (*rti.connexdds.StringPairSeq* method), 574
- `__setitem__()` (*rti.connexdds.StringSeq* method), 576
- `__setitem__()` (*rti.connexdds.SubscriberSeq* method), 583
- `__setitem__()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 587
- `__setitem__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 597
- `__setitem__()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 608
- `__setitem__()` (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 624
- `__setitem__()` (*rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq* method), 626
- `__setitem__()` (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 617
- `__setitem__()` (*rti.connexdds.ThreadSettingsKindMask* method), 635
- `__setitem__()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method),

642

`__setitem__()` (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq method*), 652

`__setitem__()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method*), 663

`__setitem__()` (*rti.connexdds.TopicBuiltinTopicDataSeq method*), 677

`__setitem__()` (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq method*), 679

`__setitem__()` (*rti.connexdds.TopicBuiltinTopicData.TopicSeq method*), 672

`__setitem__()` (*rti.connexdds.TopicSeq method*), 691

`__setitem__()` (*rti.connexdds.TransportBuiltinMask method*), 694

`__setitem__()` (*rti.connexdds.TransportInfoSeq method*), 700

`__setitem__()` (*rti.connexdds.TransportInfoVector method*), 702

`__setitem__()` (*rti.connexdds.TransportMulticastSeq method*), 707

`__setitem__()` (*rti.connexdds.TransportMulticastSettingsSeq method*), 710

`__setitem__()` (*rti.connexdds.TransportUnicastSettingsSeq method*), 715

`__setitem__()` (*rti.connexdds.Uint8Seq method*), 735

`__setitem__()` (*rti.connexdds.Uint16Seq method*), 727

`__setitem__()` (*rti.connexdds.Uint32Seq method*), 730

`__setitem__()` (*rti.connexdds.Uint64Seq method*), 733

`__setitem__()` (*rti.connexdds.UnionMemberSeq method*), 739

`__setitem__()` (*rti.connexdds.ViewState method*), 746

`__setitem__()` (*rti.connexdds.WcharSeq method*), 750

`__setitem__()` (*rti.connexdds.WstringSeq method*), 758

`__setstate__()` (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind method*), 36

`__setstate__()` (*rti.connexdds.CdrPaddingKind.CdrPaddingKind method*), 62

`__setstate__()` (*rti.connexdds.CharSeq method*), 67

`__setstate__()` (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 93

`__setstate__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 136

`__setstate__()` (*rti.connexdds.DestinationOrderKind.DestinationOrderKind method*), 143

`__setstate__()` (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 145

`__setstate__()` (*rti.connexdds.DurabilityKind.DurabilityKind method*), 180

`__setstate__()` (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 246

`__setstate__()` (*rti.connexdds.ExtensibilityKind.ExtensibilityKind method*), 270

`__setstate__()` (*rti.connexdds.Float32Seq method*), 276

`__setstate__()` (*rti.connexdds.Float64Seq method*), 278

`__setstate__()` (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 283

`__setstate__()` (*rti.connexdds.HistoryKind.HistoryKind method*), 289

`__setstate__()` (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 306

`__setstate__()` (*rti.connexdds.InstanceStateConsistencyKind method*), 314

`__setstate__()` (*rti.connexdds.Int8Seq method*), 324

`__setstate__()` (*rti.connexdds.Int16Seq method*), 316

`__setstate__()` (*rti.connexdds.Int32Seq method*), 318

`__setstate__()` (*rti.connexdds.Int64Seq method*), 322

`__setstate__()` (*rti.connexdds.LivelinessKind.LivelinessKind method*), 329

`__setstate__()` (*rti.connexdds LocatorKind.LocatorKind method*), 337

`__setstate__()` (*rti.connexdds.LogCategory.LogCategory method*), 342

`__setstate__()` (*rti.connexdds.OwnershipKind.OwnershipKind method*), 368

`__setstate__()` (*rti.connexdds.PersistentJournalKind method*), 413

`__setstate__()` (*rti.connexdds.PersistentSynchronizationKind method*), 414

`__setstate__()` (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind method*), 417

`__setstate__()` (*rti.connexdds.PrintFormatKind.PrintFormatKind method*), 421

`__setstate__()` (*rti.connexdds.PrintFormat.PrintFormat method*), 419

`__setstate__()` (*rti.connexdds.PublishModeKind.PublishModeKind method*), 471

`__setstate__()` (*rti.connexdds.ReliabilityKind.ReliabilityKind method*), 491

`__setstate__()` (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind method*), 494

`__setstate__()` (*rti.connexdds.ServiceKind.ServiceKind method*), 522

`__setstate__()`

(*rti.connexdds.ServiceRequestId.ServiceRequestId method*), 559

`__setstate__()` (*rti.connexdds.SyslogVerbosity method*), 629

`__setstate__()` (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind method*), 632

`__setstate__()` (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind method*), 689

`__setstate__()` (*rti.connexdds.TransportClassId.TransportClassId method*), 698

`__setstate__()` (*rti.connexdds.TransportMulticastKind.TransportMulticastKind method*), 704

`__setstate__()` (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind method*), 718

`__setstate__()` (*rti.connexdds.TypeKind.TypeKind method*), 724

`__setstate__()` (*rti.connexdds.Uint8Seq method*), 735

`__setstate__()` (*rti.connexdds.Uint16Seq method*), 728

`__setstate__()` (*rti.connexdds.Uint32Seq method*), 730

`__setstate__()` (*rti.connexdds.Uint64Seq method*), 733

`__setstate__()` (*rti.connexdds.Verbosity.Verbosity method*), 743

`__setstate__()` (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind method*), 754

`__str__()` (*rti.connexdds.AcknowledgmentKind method*), 37

`__str__()` (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind method*), 36

`__str__()` (*rti.connexdds.ActivityContextMask method*), 39

`__str__()` (*rti.connexdds.CdrPaddingKind method*), 63

`__str__()` (*rti.connexdds.CdrPaddingKind.CdrPaddingKind method*), 62

`__str__()` (*rti.connexdds.CoherentSetInfo method*), 69

`__str__()` (*rti.connexdds.CompressionIdMask method*), 71

`__str__()` (*rti.connexdds.DataReaderInstanceRemovalKind method*), 94

`__str__()` (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind method*), 93

`__str__()` (*rti.connexdds.DataReaderQos method*), 102

`__str__()` (*rti.connexdds.DataWriterQos method*), 130

`__str__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind method*), 137

`__str__()` (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind method*), 136

`__str__()` (*rti.connexdds.DestinationOrderKind method*), 144

`__str__()` (*rti.connexdds.DestinationOrderKind.DestinationOrderKind method*), 143

`__str__()` (*rti.connexdds.DestinationOrderScopeKind method*), 146

`__str__()` (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind method*), 145

`__str__()` (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method*), 152

`__str__()` (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method*), 155

`__str__()` (*rti.connexdds.DomainParticipantFactoryQos method*), 165

`__str__()` (*rti.connexdds.DomainParticipantQos method*), 170

`__str__()` (*rti.connexdds.DurabilityKind method*), 181

`__str__()` (*rti.connexdds.DurabilityKind.DurabilityKind method*), 180

`__str__()` (*rti.connexdds.DynamicData method*), 221

`__str__()` (*rti.connexdds.DynamicDataEncapsulationKind method*), 247

`__str__()` (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind method*), 246

`__str__()` (*rti.connexdds.DynamicType method*), 255

`__str__()` (*rti.connexdds.EnumMember method*), 263

`__str__()` (*rti.connexdds.ExtensibilityKind method*), 271

`__str__()` (*rti.connexdds.ExtensibilityKind.ExtensibilityKind method*), 270

`__str__()` (*rti.connexdds.FlowControllerSchedulingPolicy method*), 284

`__str__()` (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy method*), 283

`__str__()` (*rti.connexdds.Guid method*), 287

`__str__()` (*rti.connexdds.HistoryKind method*), 290

`__str__()` (*rti.connexdds.HistoryKind.HistoryKind method*), 289

`__str__()` (*rti.connexdds.IgnoredEntityReplacementKind method*), 307

`__str__()` (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind method*), 306

`__str__()` (*rti.connexdds.InstanceHandle method*), 308

`__str__()` (*rti.connexdds.InstanceState method*), 312

`__str__()` (*rti.connexdds.InstanceStateConsistencyKind method*), 314

`__str__()` (*rti.connexdds.LivelinessKind method*), 330

`__str__()` (*rti.connexdds.LivelinessKind.LivelinessKind method*), 329

`__str__()` (*rti.connexdds.LocatorKind method*), 338

`__str__()` (*rti.connexdds.LocatorKind.LocatorKind method*), 337

`__str__()` (*rti.connexdds.LogCategory method*), 343

`__str__()` (*rti.connexdds.LogCategory.LogCategory method*), 342

`__str__()` (*rti.connexdds.LongDouble method*), 345

`__str__()` (*rti.connexdds.OwnershipKind method*), 369

`__str__()` (*rti.connexdds.OwnershipKind.OwnershipKind method*), 368

- `__str__` () (*rti.connexdds.PersistentJournalKind* method), 413
`__str__` () (*rti.connexdds.PersistentSynchronizationKind* method), 414
`__str__` () (*rti.connexdds.PresentationAccessScopeKind* method), 418
`__str__` () (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind* method), 417
`__str__` () (*rti.connexdds.PrintFormat* method), 420
`__str__` () (*rti.connexdds.PrintFormatKind* method), 422
`__str__` () (*rti.connexdds.PrintFormatKind.PrintFormatKind* method), 421
`__str__` () (*rti.connexdds.PrintFormat.PrintFormat* method), 419
`__str__` () (*rti.connexdds.ProductVersion* method), 423
`__str__` () (*rti.connexdds.PublisherQos* method), 475
`__str__` () (*rti.connexdds.PublishModeKind* method), 472
`__str__` () (*rti.connexdds.PublishModeKind.PublishModeKind* method), 471
`__str__` () (*rti.connexdds.ReliabilityKind* method), 492
`__str__` () (*rti.connexdds.ReliabilityKind.ReliabilityKind* method), 491
`__str__` () (*rti.connexdds.RemoteParticipantPurgeKind* method), 495
`__str__` () (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* method), 494
`__str__` () (*rti.connexdds.RtpsReservedPortKindMask* method), 502
`__str__` () (*rti.connexdds.SampleFlag* method), 506
`__str__` () (*rti.connexdds.SampleIdentity* method), 508
`__str__` () (*rti.connexdds.SampleLostState* method), 512
`__str__` () (*rti.connexdds.SampleRejectedState* method), 515
`__str__` () (*rti.connexdds.SampleState* method), 518
`__str__` () (*rti.connexdds.ServiceKind* method), 523
`__str__` () (*rti.connexdds.ServiceKind.ServiceKind* method), 522
`__str__` () (*rti.connexdds.ServiceRequestId* method), 560
`__str__` () (*rti.connexdds.ServiceRequestId.ServiceRequestId* method), 559
`__str__` () (*rti.connexdds.StatusMask* method), 568
`__str__` () (*rti.connexdds.StreamKind* method), 571
`__str__` () (*rti.connexdds.SubscriberQos* method), 582
`__str__` () (*rti.connexdds.SyslogVerbosity* method), 629
`__str__` () (*rti.connexdds.ThreadSettingsCpuRotationKind* method), 633
`__str__` () (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* method), 632
`__str__` () (*rti.connexdds.ThreadSettingsKindMask* method), 635
`__str__` () (*rti.connexdds.TopicQos* method), 685
`__str__` () (*rti.connexdds.TopicQuerySelectionKind* method), 690
`__str__` () (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind* method), 689
`__str__` () (*rti.connexdds.TransportBuiltinMask* method), 694
`__str__` () (*rti.connexdds.TransportClassId* method), 699
`__str__` () (*rti.connexdds.TransportClassId.TransportClassId* method), 698
`__str__` () (*rti.connexdds.TransportMulticastKind* method), 705
`__str__` () (*rti.connexdds.TransportMulticastKind.TransportMulticastKind* method), 704
`__str__` () (*rti.connexdds.TypeConsistencyEnforcementKind* method), 720
`__str__` () (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind* method), 719
`__str__` () (*rti.connexdds.TypeKind* method), 725
`__str__` () (*rti.connexdds.TypeKind.TypeKind* method), 724
`__str__` () (*rti.connexdds.Verbosity* method), 744
`__str__` () (*rti.connexdds.Verbosity.Verbosity* method), 743
`__str__` () (*rti.connexdds.ViewState* method), 746
`__str__` () (*rti.connexdds.WireProtocolAutoKind* method), 755
`__str__` () (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* method), 754
`__sub__` () (*rti.connexdds.Duration* method), 183
`__sub__` () (*rti.connexdds.SequenceNumber* method), 520
`__sub__` () (*rti.connexdds.Time* method), 637
`__truediv__` () (*rti.connexdds.Duration* method), 183
`__weakref__` (*rti.connexdds.Exception* attribute), 268
`__weakref__` (*rti.rpc.Replier* attribute), 765
`__weakref__` (*rti.rpc.Requester* attribute), 761
`__weakref__` (*rti.rpc.SimpleReplier* attribute), 768
`__xor__` () (*rti.connexdds.ActivityContextMask* method), 39
`__xor__` () (*rti.connexdds.CompressionIdMask* method), 71
`__xor__` () (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 152
`__xor__` () (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 155
`__xor__` () (*rti.connexdds.InstanceState* method), 312
`__xor__` () (*rti.connexdds.RtpsReservedPortKindMask* method), 502
`__xor__` () (*rti.connexdds.SampleFlag* method), 506
`__xor__` () (*rti.connexdds.SampleLostState* method), 512
`__xor__` () (*rti.connexdds.SampleRejectedState* method), 515
`__xor__` () (*rti.connexdds.SampleState* method), 518
`__xor__` () (*rti.connexdds.StatusMask* method), 568
`__xor__` () (*rti.connexdds.StreamKind* method), 571
`__xor__` () (*rti.connexdds.ThreadSettingsKindMask* method), 635
`__xor__` () (*rti.connexdds.TransportBuiltinMask* method), 694
`__xor__` () (*rti.connexdds.ViewState* method), 746
- A**
- `absolute_generation` (*rti.connexdds.Rank* property), 487
`accept_unknown_peers` (*rti.connexdds.Discovery* property), 147
`access_scope` (*rti.connexdds.Presentation* property), 415

- acknowledge_all() (*rti.connexdds.DataReader method*), 86
- acknowledge_all() (*rti.connexdds.DynamicData.DataReader method*), 191
- acknowledge_all() (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*), 376
- acknowledge_all() (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 432
- acknowledge_all() (*rti.connexdds.ServiceRequest.DataReader method*), 530
- acknowledge_all() (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*), 591
- acknowledge_all() (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 646
- acknowledge_sample() (*rti.connexdds.DataReader method*), 87
- acknowledge_sample() (*rti.connexdds.DynamicData.DataReader method*), 191
- acknowledge_sample() (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*), 376
- acknowledge_sample() (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 433
- acknowledge_sample() (*rti.connexdds.ServiceRequest.DataReader method*), 530
- acknowledge_sample() (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*), 591
- acknowledge_sample() (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 646
- acknowledgment_kind (*rti.connexdds.Reliability property*), 490
- AcknowledgmentInfo (*class in rti.connexdds*), 34
- AcknowledgmentKind (*class in rti.connexdds*), 35
- AcknowledgmentKind.AcknowledgmentKind (*class in rti.connexdds*), 35
- AckResponseData (*class in rti.connexdds*), 34
- ACTEnumMember (*class in rti.connexdds*), 31
- active_count (*rti.connexdds.ReliableReaderActivityChangedStatus property*), 492
- ActivityContextMask (*class in rti.connexdds*), 37
- ACTMember (*class in rti.connexdds*), 32
- ACTUnionMember (*class in rti.connexdds*), 33
- add_cookie() (*rti.connexdds.DynamicData.WriterContentFilterHelper method*), 220
- add_cookie() (*rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilterHelper method*), 405
- add_cookie() (*rti.connexdds.PublicationBuiltinTopicData.WriterContentFilterHelper method*), 461
- add_cookie() (*rti.connexdds.ServiceRequest.WriterContentFilterHelper method*), 557
- add_cookie() (*rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper method*), 620
- add_cookie() (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilterHelper method*), 674
- add_member() (*rti.connexdds.EnumType method*), 266
- add_member() (*rti.connexdds.StructType method*), 578
- add_members() (*rti.connexdds.EnumType method*), 266
- add_members() (*rti.connexdds.StructType method*), 578
- add_members() (*rti.connexdds.UnionType method*), 740
- add_parameter() (*rti.connexdds.Query method*), 485
- add_parameters() (*rti.connexdds.Filter method*), 271
- add_peer() (*rti.connexdds.DomainParticipant method*), 158
- add_peers() (*rti.connexdds.DomainParticipant method*), 158
- address (*rti.connexdds.Locator property*), 331
- addresses (*rti.connexdds.MulticastMapping property*), 358
- AGGREGATION_TYPE (*rti.connexdds.TypeKind attribute*), 720
- AGGREGATION_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722
- ALERT (*rti.connexdds.SyslogVerbosity attribute*), 628
- alert() (*rti.connexdds.Logger method*), 344
- ALIAS_TYPE (*rti.connexdds.TypeKind attribute*), 720
- ALIAS_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722
- AliasType (*class in rti.connexdds*), 40
- ALIVE (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind attribute*), 134
- ALIVE (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute*), 135
- ALIVE (*rti.connexdds.InstanceState attribute*), 311
- alive_count (*rti.connexdds.LivelinessChangedStatus property*), 328
- alive_count_change (*rti.connexdds.LivelinessChangedStatus property*), 328
- alive_instance_count (*rti.connexdds.DataReaderCacheStatus property*), 90
- alive_instance_count (*rti.connexdds.DataWriterCacheStatus property*), 120
- alive_instance_count_peak (*rti.connexdds.DataReaderCacheStatus property*), 90
- alive_instance_count_peak (*rti.connexdds.DataWriterCacheStatus property*), 120
- alive_instance_removal (*rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings property*), 107

- ALIVE_OR_DISPOSED (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind* attribute), 134
- ALIVE_OR_DISPOSED (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind* attribute), 135
- ALIVE_THEN_DISPOSED (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind* attribute), 134
- ALIVE_THEN_DISPOSED (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind* attribute), 135
- ALL (*rti.connexdds.ActivityContextMask* attribute), 37
- ALL (*rti.connexdds.CompressionIdMask* attribute), 70
- ALL (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* attribute), 151
- ALL (*rti.connexdds.RtpsReservedPortKindMask* attribute), 500
- ALL (*rti.connexdds.StatusMask* attribute), 566
- all (*rti.connexdds.TransportBuiltin* attribute), 692
- ALL (*rti.connexdds.TransportBuiltinMask* attribute), 693
- all_categories (*rti.connexdds.LogCategory* attribute), 343
- all_categories (*rti.connexdds.LogCategory.LogCategory* attribute), 342
- AllocationSettings (*class in rti.connexdds*), 41
- allow_type_coercion (*rti.connexdds.TypeConsistencyEnforcement* attribute), 717
- ALLOW_TYPE_COERCION (*rti.connexdds.TypeConsistencyEnforcementKind* attribute), 717
- ALLOW_TYPE_COERCION (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind* attribute), 718
- AlreadyClosedError, 42
- ANNOTATION_TYPE (*rti.connexdds.TypeKind* attribute), 720
- ANNOTATION_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 722
- any (*rti.connexdds.DataState* attribute), 111
- any (*rti.connexdds.DataStateEx* attribute), 112
- ANY (*rti.connexdds.InstanceState* attribute), 311
- ANY (*rti.connexdds.LocatorKind* attribute), 335
- ANY (*rti.connexdds.LocatorKind.LocatorKind* attribute), 336
- ANY (*rti.connexdds.SampleState* attribute), 516
- ANY (*rti.connexdds.StreamKind* attribute), 569
- ANY (*rti.connexdds.TransportClassId* attribute), 696
- ANY (*rti.connexdds.TransportClassId.TransportClassId* attribute), 696
- ANY (*rti.connexdds.ViewState* attribute), 744
- any_data (*rti.connexdds.DataState* attribute), 111
- any_data (*rti.connexdds.DataStateEx* attribute), 112
- ANY_INSTANCE (*rti.connexdds.DataReaderInstanceRemovalKind* attribute), 91
- ANY_INSTANCE (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind* attribute), 92
- AnyDataReader (*class in rti.connexdds*), 42
- AnyDataReaderListener (*class in rti.connexdds*), 42
- AnyDataReaderSeq (*class in rti.connexdds*), 43
- AnyDataWriter (*class in rti.connexdds*), 45
- AnyDataWriterListener (*class in rti.connexdds*), 45
- AnyDataWriterSeq (*class in rti.connexdds*), 46
- AnyTopic (*class in rti.connexdds*), 48
- AnyTopicListener (*class in rti.connexdds*), 48
- AnyTopicSeq (*class in rti.connexdds*), 49
- api (*rti.connexdds.LogCategory* attribute), 343
- api (*rti.connexdds.LogCategory.LogCategory* attribute), 342
- app_ack_period (*rti.connexdds.RtpsReliableReaderProtocol* property), 497
- append () (*rti.connexdds.AnyDataReaderSeq* method), 44
- append () (*rti.connexdds.AnyDataWriterSeq* method), 47
- append () (*rti.connexdds.AnyTopicSeq* method), 50
- append () (*rti.connexdds.BoolSeq* method), 55
- append () (*rti.connexdds.ChannelSettingsSeq* method), 65
- append () (*rti.connexdds.CharSeq* method), 67
- append () (*rti.connexdds.ConditionSeq* method), 75
- append () (*rti.connexdds.ContentFilteredTopicSeq* method), 79
- append () (*rti.connexdds.CookieSeq* method), 81
- append () (*rti.connexdds.DataReaderSeq* method), 108
- append () (*rti.connexdds.DataWriterSeq* method), 139
- append () (*rti.connexdds.DomainParticipantSeq* method), 177
- append () (*rti.connexdds.DynamicData* method), 221
- append () (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 187
- append () (*rti.connexdds.DynamicData.DataReaderSeq* method), 196
- append () (*rti.connexdds.DynamicData.DataWriterSeq* method), 207
- append () (*rti.connexdds.DynamicDataSeq* method), 251
- append () (*rti.connexdds.DynamicDataTimestampedSeq* method), 253
- append () (*rti.connexdds.DynamicData.TopicSeq* method), 217
- append () (*rti.connexdds.EndpointGroupSeq* method), 258
- append () (*rti.connexdds.EntitySeq* method), 262
- append () (*rti.connexdds.EnumMemberSeq* method), 265
- append () (*rti.connexdds.Float32Seq* method), 276
- append () (*rti.connexdds.Float64Seq* method), 278
- append () (*rti.connexdds.Float128Seq* method), 273
- append () (*rti.connexdds.IAnyDataReaderSeq* method), 293
- append () (*rti.connexdds.IAnyDataWriterSeq* method), 296
- append () (*rti.connexdds.IAnyTopicSeq* method), 298
- append () (*rti.connexdds.IConditionSeq* method), 301
- append () (*rti.connexdds.IEntitySeq* method), 304
- append () (*rti.connexdds.InstanceHandleSeq* method), 310
- append () (*rti.connexdds.Int8Seq* method), 324
- append () (*rti.connexdds.Int16Seq* method), 316
- append () (*rti.connexdds.Int32Seq* method), 318
- append () (*rti.connexdds.Int64Seq* method), 322
- append () (*rti.connexdds.LocatorFilterElementSeq* method), 334
- append () (*rti.connexdds.LocatorSeq* method), 339
- append () (*rti.connexdds.MemberSeq* method), 348
- append () (*rti.connexdds.MonitoringMetricSelectionSeq* method), 355

- append () (*rti.connexdds.MulticastMappingSeq* method), 360
- append () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 373
- append () (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 382
- append () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 394
- append () (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 408
- append () (*rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq* method), 410
- append () (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 403
- append () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 429
- append () (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 438
- append () (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 450
- append () (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 465
- append () (*rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq* method), 468
- append () (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 459
- append () (*rti.connexdds.PublisherSeq* method), 477
- append () (*rti.connexdds.QosPolicyCountSeq* method), 480
- append () (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq* method), 526
- append () (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 535
- append () (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 546
- append () (*rti.connexdds.ServiceRequestSeq* method), 562
- append () (*rti.connexdds.ServiceRequestTimestampedSeq* method), 564
- append () (*rti.connexdds.ServiceRequest.TopicSeq* method), 555
- append () (*rti.connexdds.StringPairSeq* method), 574
- append () (*rti.connexdds.StringSeq* method), 576
- append () (*rti.connexdds.SubscriberSeq* method), 583
- append () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 587
- append () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 597
- append () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 609
- append () (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 624
- append () (*rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq* method), 626
- append () (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq* method), 618
- append () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq* method), 643
- append () (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq* method), 652
- append () (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq* method), 663
- append () (*rti.connexdds.TopicBuiltinTopicDataSeq* method), 677
- append () (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq* method), 680
- append () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq* method), 672
- append () (*rti.connexdds.TopicSeq* method), 691
- append () (*rti.connexdds.TransportInfoSeq* method), 701
- append () (*rti.connexdds.TransportMulticastSeq* method), 707
- append () (*rti.connexdds.TransportMulticastSettingsSeq* method), 710
- append () (*rti.connexdds.TransportUnicastSettingsSeq* method), 715
- append () (*rti.connexdds.Uint8Seq* method), 735
- append () (*rti.connexdds.Uint16Seq* method), 728
- append () (*rti.connexdds.Uint32Seq* method), 730
- append () (*rti.connexdds.Uint64Seq* method), 733
- append () (*rti.connexdds.UnionMemberSeq* method), 739
- append () (*rti.connexdds.WcharSeq* method), 751
- append () (*rti.connexdds.WstringSeq* method), 758
- append_to_expression_parameter () (*rti.connexdds.ContentFilteredTopic* method), 77
- append_to_expression_parameter () (*rti.connexdds.DynamicData.ContentFilteredTopic* method), 185
- append_to_expression_parameter () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic* method), 370
- append_to_expression_parameter () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic* method), 427
- append_to_expression_parameter () (*rti.connexdds.ServiceRequest.ContentFilteredTopic* method), 524
- append_to_expression_parameter () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic* method), 585

append_to_expression_parameter() (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic method*), 641
 APPLICATION_AUTO (*rti.connexdds.AcknowledgmentKind attribute*), 35
 APPLICATION_AUTO (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind attribute*), 35
 APPLICATION_EXPLICIT (*rti.connexdds.AcknowledgmentKind attribute*), 35
 APPLICATION_EXPLICIT (*rti.connexdds.AcknowledgmentKind attribute*), 36
 application_kind (*rti.logging.distlog.LoggerOptions property*), 772
 application_name (*rti.connexdds.Monitoring property*), 349
 APPLICATION_ORDERED (*rti.connexdds.AcknowledgmentKind attribute*), 35
 APPLICATION_ORDERED (*rti.connexdds.AcknowledgmentKind attribute*), 36
 ARRAY_TYPE (*rti.connexdds.TypeKind attribute*), 720
 ARRAY_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722
 ArrayType (*class in rti.connexdds*), 51
 assert_liveliness() (*rti.connexdds.DataWriter method*), 115
 assert_liveliness() (*rti.connexdds.DomainParticipant method*), 158
 assert_liveliness() (*rti.connexdds.DynamicData.DataWriter method*), 199
 assert_liveliness() (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*), 385
 assert_liveliness() (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*), 441
 assert_liveliness() (*rti.connexdds.ServiceRequest.DataWriter method*), 538
 assert_liveliness() (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method*), 599
 assert_liveliness() (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 654
 assertions_per_lease_duration (*rti.connexdds.Liveliness property*), 327
 ASYNCHRONOUS (*rti.connexdds.PublishModeKind attribute*), 470
 ASYNCHRONOUS (*rti.connexdds.PublishModeKind.PublishModeKind attribute*), 470
 asynchronous() (*rti.connexdds.PublishMode static method*), 469
 asynchronous_batch_thread (*rti.connexdds.AsynchronousPublisher property*), 52
 asynchronous_publisher (*rti.connexdds.DiscoveryConfig property*), 148
 asynchronous_publisher (*rti.connexdds.PublisherQos property*), 475
 AsynchronousPublisher (*class in rti.connexdds*), 51
 attach_condition() (*rti.connexdds.WaitSet method*), 748
 AUTO (*rti.connexdds.CdrPaddingKind attribute*), 61
 AUTO (*rti.connexdds.CdrPaddingKind.CdrPaddingKind attribute*), 61
 AUTO_COUNT (*rti.connexdds.AllocationSettings attribute*), 41
 auto_enable (*rti.connexdds.EntityFactory attribute*), 260
 AUTO_ID (*rti.connexdds.DataRepresentation attribute*), 109
 AUTO_MAX_TOTAL_INSTANCES (*rti.connexdds.DataReaderResourceLimits attribute*), 104
 auto_purge_disposed_samples() (*rti.connexdds.ReaderDataLifecycle static method*), 488
 auto_purge_no_writer_samples() (*rti.connexdds.ReaderDataLifecycle static method*), 488
 auto_type_coercion (*rti.connexdds.TypeConsistencyEnforcement attribute*), 717
 AUTO_TYPE_COERCION (*rti.connexdds.TypeConsistencyEnforcementKind attribute*), 717
 AUTO_TYPE_COERCION (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind attribute*), 718
 AUTO_WRITER_DEPTH (*rti.connexdds.Durability attribute*), 178
 autodispose_unregistered_instances (*rti.connexdds.WriterDataLifecycle property*), 756
 autoenable_created_entities (*rti.connexdds.EntityFactory property*), 260
 automatic (*rti.connexdds.Duration attribute*), 183
 automatic (*rti.connexdds.Guid attribute*), 288
 automatic (*rti.connexdds.Liveliness attribute*), 327
 AUTOMATIC (*rti.connexdds.LivelinessKind attribute*), 328
 AUTOMATIC (*rti.connexdds.LivelinessKind.LivelinessKind attribute*), 329
 automatic (*rti.connexdds.SampleIdentity attribute*), 508
 automatic (*rti.connexdds.SequenceNumber attribute*), 520
 AUTOMATIC (*rti.connexdds.TransportMulticastKind attribute*), 703
 AUTOMATIC (*rti.connexdds.TransportMulticastKind.TransportMulticastKind attribute*), 703
 autopurge_disposed_instances_delay (*rti.connexdds.ReaderDataLifecycle property*), 488
 autopurge_disposed_instances_delay (*rti.connexdds.WriterDataLifecycle property*), 756
 autopurge_disposed_samples_delay (*rti.connexdds.ReaderDataLifecycle property*), 488
 autopurge_nowriter_instances_delay (*rti.connexdds.ReaderDataLifecycle property*), 488
 autopurge_nowriter_samples_delay (*rti.connexdds.ReaderDataLifecycle property*), 488

- autopurge_remote_not_alive_writer_delay
(*rti.connexdds.DataReaderResourceLimits* property), 104
 - autopurge_remote_virtual_writer_delay
(*rti.connexdds.DataReaderResourceLimits* property), 104
 - autopurge_unregistered_instances_delay
(*rti.connexdds.WriterDataLifecycle* property), 756
 - autoregister_instances
(*rti.connexdds.DataWriterResourceLimits* property), 133
 - Availability (*class in rti.connexdds*), 52
 - availability (*rti.connexdds.DataReaderQos* property), 102
 - availability (*rti.connexdds.DataWriterQos* property), 130
- ## B
- backwards_compatible
(*rti.connexdds.RtpsWellKnownPorts* attribute), 504
 - banish_ignored_participants()
(*rti.connexdds.DomainParticipant* method), 158
 - baseline (*rti.connexdds.BuiltinProfiles* attribute), 56
 - baseline_5_0_0 (*rti.connexdds.BuiltinProfiles* attribute), 56
 - baseline_5_1_0 (*rti.connexdds.BuiltinProfiles* attribute), 56
 - baseline_5_2_0 (*rti.connexdds.BuiltinProfiles* attribute), 56
 - baseline_5_3_0 (*rti.connexdds.BuiltinProfiles* attribute), 56
 - baseline_6_0_0 (*rti.connexdds.BuiltinProfiles* attribute), 56
 - Batch (*class in rti.connexdds*), 53
 - batch (*rti.connexdds.DataWriterQos* property), 130
 - best_effort (*rti.connexdds.Reliability* attribute), 490
 - BEST_EFFORT (*rti.connexdds.ReliabilityKind* attribute), 490
 - BEST_EFFORT (*rti.connexdds.ReliabilityKind.ReliabilityKind* attribute), 491
 - bitset (*rti.connexdds.Member* property), 346
 - BITSET_TYPE (*rti.connexdds.TypeKind* attribute), 720
 - BITSET_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 722
 - BOOLEAN_TYPE (*rti.connexdds.TypeKind* attribute), 720
 - BOOLEAN_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 722
 - BoolSeq (*class in rti.connexdds*), 54
 - BoolType (*class in rti.connexdds*), 56
 - bounds (*rti.connexdds.UnidimensionalCollectionType* property), 737
 - buffer_alignment (*rti.connexdds.ReceiverPool* property), 489
 - buffer_size (*rti.connexdds.ReceiverPool* property), 489
 - builtin_discovery_plugins
(*rti.connexdds.DiscoveryConfig* property), 148
 - BUILTIN_MULTICAST
(*rti.connexdds.RtpsReservedPortKindMask* attribute), 500
 - builtin_multicast_port_offset
(*rti.connexdds.RtpsWellKnownPorts* property), 504
 - builtin_subscriber (*rti.connexdds.DomainParticipant* property), 158
 - builtin_topic_name
(*rti.connexdds.PublicationBuiltinTopicData* attribute), 462
 - builtin_topic_name
(*rti.connexdds.SubscriptionBuiltinTopicData* attribute), 620
 - BUILTIN_UNICAST
(*rti.connexdds.RtpsReservedPortKindMask* attribute), 500
 - builtin_unicast_port_offset
(*rti.connexdds.RtpsWellKnownPorts* property), 504
 - BuiltinProfiles (*class in rti.connexdds*), 56
 - BuiltinTopicKey (*class in rti.connexdds*), 58
 - BuiltinTopicReaderResourceLimits (*class in rti.connexdds*), 59
 - BY_RECEPTION_TIMESTAMP
(*rti.connexdds.DestinationOrderKind* attribute), 142
 - BY_RECEPTION_TIMESTAMP (*rti.connexdds.DestinationOrderKind.DestinationOrderKind* attribute), 143
 - BY_SOURCE_TIMESTAMP
(*rti.connexdds.DestinationOrderKind* attribute), 142
 - BY_SOURCE_TIMESTAMP (*rti.connexdds.DestinationOrderKind.DestinationOrderKind* attribute), 143
 - Bytes (*class in rti.types.builtin*), 759
 - bytes_per_token
(*rti.connexdds.FlowControllerTokenBucketProperty* property), 285
 - ByteVector (*class in rti.connexdds*), 60
 - BZIP2 (*rti.connexdds.CompressionIdMask* attribute), 70
- ## C
- CANCEL_ASYNCHRONOUS
(*rti.connexdds.ThreadSettingsKindMask* attribute), 633
 - category (*rti.logging.distlog.MessageParams* property), 773
 - CDR_BIG_ENDIAN
(*rti.connexdds.DynamicDataEncapsulationKind* attribute), 245
 - CDR_BIG_ENDIAN (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind* attribute), 245
 - CDR_LITTLE_ENDIAN
(*rti.connexdds.DynamicDataEncapsulationKind* attribute), 245
 - CDR_LITTLE_ENDIAN (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind* attribute), 245
 - cdr_padding_kind (*rti.connexdds.TypeSupport* property), 726
 - cdr_serialized_sample_max_size()
(*rti.connexdds.ACTEnumMember* method), 31
 - cdr_serialized_sample_max_size()
(*rti.connexdds.ACTMember* method), 33
 - cdr_serialized_sample_max_size()
(*rti.connexdds.ACTUnionMember* method), 34
 - cdr_serialized_sample_min_size()
(*rti.connexdds.ACTEnumMember* method), 32
 - cdr_serialized_sample_min_size()
(*rti.connexdds.ACTMember* method), 33

- `cdr_serialized_sample_min_size()`
(*rti.connexdds.ACTUnionMember* method), 34
- `CdrPaddingKind` (class in *rti.connexdds*), 61
- `CdrPaddingKind.CdrPaddingKind` (class in *rti.connexdds*), 61
- `change` (*rti.connexdds.EventCount32* property), 267
- `change` (*rti.connexdds.EventCount64* property), 267
- `channel_filter_expression_max_length`
(*rti.connexdds.DomainParticipantResourceLimits* property), 171
- `channel_seq_max_length`
(*rti.connexdds.DomainParticipantResourceLimits* property), 171
- `channels` (*rti.connexdds.MultiChannel* property), 357
- `ChannelSettings` (class in *rti.connexdds*), 63
- `ChannelSettingsSeq` (class in *rti.connexdds*), 63
- `CHAR_8_TYPE` (*rti.connexdds.TypeKind* attribute), 720
- `CHAR_8_TYPE` (*rti.connexdds.TypeKind.TypeKind* attribute), 722
- `CHAR_16_TYPE` (*rti.connexdds.TypeKind* attribute), 720
- `CHAR_16_TYPE` (*rti.connexdds.TypeKind.TypeKind* attribute), 722
- `CharSeq` (class in *rti.connexdds*), 65
- `CharType` (in module *rti.connexdds*), 68
- `check_buffer_size` (*rti.connexdds.DynamicDataProperty* property), 249
- `check_crc` (*rti.connexdds.WireProtocol* property), 752
- `class_id` (*rti.connexdds.TransportInfo* property), 699
- `cleanup_period` (*rti.connexdds.Database* property), 141
- `clear()` (*rti.connexdds.AnyDataReaderSeq* method), 44
- `clear()` (*rti.connexdds.AnyDataWriterSeq* method), 48
- `clear()` (*rti.connexdds.AnyTopicSeq* method), 50
- `clear()` (*rti.connexdds.BoolSeq* method), 55
- `clear()` (*rti.connexdds.ByteVector* method), 61
- `clear()` (*rti.connexdds.ChannelSettingsSeq* method), 65
- `clear()` (*rti.connexdds.CharSeq* method), 67
- `clear()` (*rti.connexdds.ConditionSeq* method), 75
- `clear()` (*rti.connexdds.ContentFilteredTopicSeq* method), 79
- `clear()` (*rti.connexdds.CookieSeq* method), 81
- `clear()` (*rti.connexdds.CookieVector* method), 83
- `clear()` (*rti.connexdds.DataReaderSeq* method), 108
- `clear()` (*rti.connexdds.DataWriterSeq* method), 139
- `clear()` (*rti.connexdds.DomainParticipantSeq* method), 177
- `clear()` (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 187
- `clear()` (*rti.connexdds.DynamicData.DataReaderSeq* method), 196
- `clear()` (*rti.connexdds.DynamicData.DataWriterSeq* method), 207
- `clear()` (*rti.connexdds.DynamicDataSeq* method), 251
- `clear()` (*rti.connexdds.DynamicDataTimestampedSeq* method), 253
- `clear()` (*rti.connexdds.DynamicData.TopicSeq* method), 218
- `clear()` (*rti.connexdds.EndpointGroupSeq* method), 258
- `clear()` (*rti.connexdds.EndpointGroupVector* method), 259
- `clear()` (*rti.connexdds.EntitySeq* method), 262
- `clear()` (*rti.connexdds.EnumMemberSeq* method), 265
- `clear()` (*rti.connexdds.Float32Seq* method), 276
- `clear()` (*rti.connexdds.Float64Seq* method), 278
- `clear()` (*rti.connexdds.Float128Seq* method), 273
- `clear()` (*rti.connexdds.IAnyDataReaderSeq* method), 293
- `clear()` (*rti.connexdds.IAnyDataWriterSeq* method), 296
- `clear()` (*rti.connexdds.IAnyTopicSeq* method), 298
- `clear()` (*rti.connexdds.IConditionSeq* method), 301
- `clear()` (*rti.connexdds.IEntitySeq* method), 304
- `clear()` (*rti.connexdds.InstanceHandleSeq* method), 310
- `clear()` (*rti.connexdds.Int8Seq* method), 324
- `clear()` (*rti.connexdds.Int16Seq* method), 316
- `clear()` (*rti.connexdds.Int32Seq* method), 318
- `clear()` (*rti.connexdds.Int32Vector* method), 320
- `clear()` (*rti.connexdds.Int64Seq* method), 322
- `clear()` (*rti.connexdds.LocatorFilterElementSeq* method), 334
- `clear()` (*rti.connexdds.LocatorSeq* method), 340
- `clear()` (*rti.connexdds.LocatorVector* method), 341
- `clear()` (*rti.connexdds.MemberSeq* method), 348
- `clear()` (*rti.connexdds.MonitoringMetricSelectionSeq* method), 355
- `clear()` (*rti.connexdds.MulticastMappingSeq* method), 360
- `clear()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 373
- `clear()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 382
- `clear()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 394
- `clear()` (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 408
- `clear()` (*rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq* method), 410
- `clear()` (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 403
- `clear()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 429
- `clear()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 439
- `clear()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 450
- `clear()` (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 465
- `clear()` (*rti.connexdds.PublicationBuiltinTopicData.TimestampedSeq* method), 468
- `clear()` (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 459
- `clear()` (*rti.connexdds.PublisherSeq* method), 477
- `clear()` (*rti.connexdds.QosPolicyCountSeq* method), 480
- `clear()` (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq* method), 526
- `clear()` (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 535

- clear () (*rti.connexdds.ServiceRequest.DataWriterSeq method*), 546
- clear () (*rti.connexdds.ServiceRequestSeq method*), 562
- clear () (*rti.connexdds.ServiceRequestTimestampedSeq method*), 564
- clear () (*rti.connexdds.ServiceRequest.TopicSeq method*), 555
- clear () (*rti.connexdds.StringPairSeq method*), 574
- clear () (*rti.connexdds.StringSeq method*), 576
- clear () (*rti.connexdds.SubscriberSeq method*), 584
- clear () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 587
- clear () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 597
- clear () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 609
- clear () (*rti.connexdds.SubscriptionBuiltinTopicDataSeq method*), 624
- clear () (*rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 626
- clear () (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq method*), 618
- clear () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 643
- clear () (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq method*), 652
- clear () (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method*), 663
- clear () (*rti.connexdds.TopicBuiltinTopicDataSeq method*), 677
- clear () (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq method*), 680
- clear () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq method*), 672
- clear () (*rti.connexdds.TopicSeq method*), 691
- clear () (*rti.connexdds.TransportInfoSeq method*), 701
- clear () (*rti.connexdds.TransportInfoVector method*), 702
- clear () (*rti.connexdds.TransportMulticastSeq method*), 707
- clear () (*rti.connexdds.TransportMulticastSettingsSeq method*), 710
- clear () (*rti.connexdds.TransportUnicastSettingsSeq method*), 715
- clear () (*rti.connexdds.Uint8Seq method*), 735
- clear () (*rti.connexdds.Uint16Seq method*), 728
- clear () (*rti.connexdds.Uint32Seq method*), 730
- clear () (*rti.connexdds.Uint64Seq method*), 733
- clear () (*rti.connexdds.UnionMemberSeq method*), 739
- clear () (*rti.connexdds.WcharSeq method*), 751
- clear () (*rti.connexdds.WstringSeq method*), 758
- clear_all_members () (*rti.connexdds.DynamicData method*), 222
- clear_member () (*rti.connexdds.DynamicData method*), 222
- clear_optional_member () (*rti.connexdds.DynamicData method*), 222
- ClientBase (*class in rti.rpc*), 769
- close () (*rti.connexdds.DataReader method*), 87
- close () (*rti.connexdds.DataWriter method*), 115
- close () (*rti.connexdds.DynamicData.DataReader method*), 191
- close () (*rti.connexdds.DynamicData.DataWriter method*), 199
- close () (*rti.connexdds.FlowController method*), 280
- close () (*rti.connexdds.IAnyDataReader method*), 291
- close () (*rti.connexdds.IAnyDataWriter method*), 294
- close () (*rti.connexdds.IAnyTopic method*), 297
- close () (*rti.connexdds.IEntity method*), 302
- close () (*rti.connexdds.IReadCondition method*), 305
- close () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*), 377
- close () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*), 385
- close () (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 433
- close () (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*), 441
- close () (*rti.connexdds.ServiceRequest.DataReader method*), 530
- close () (*rti.connexdds.ServiceRequest.DataWriter method*), 538
- close () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*), 591
- close () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method*), 599
- close () (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 647
- close () (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 654
- close () (*rti.connexdds.TopicQuery method*), 686
- close () (*rti.logging.handler.DistlogHandler method*), 773
- close () (*rti.rpc.ClientBase method*), 769
- close () (*rti.rpc.Replier method*), 765
- close () (*rti.rpc.Requester method*), 761
- close () (*rti.rpc.Service method*), 768
- close () (*rti.rpc.SimpleReplier method*), 768
- close_contained_entities () (*rti.connexdds.DomainParticipant method*), 158
- closed (*rti.connexdds.FlowController property*), 280
- closed (*rti.connexdds.IEntity property*), 302
- closed (*rti.connexdds.IReadCondition property*), 305
- closed (*rti.connexdds.TopicQuery property*), 686
- closed (*rti.rpc.Replier property*), 765
- closed (*rti.rpc.Requester property*), 761
- closed (*rti.rpc.SimpleReplier property*), 768
- coherent_access (*rti.connexdds.Presentation property*), 415
- coherent_set_info (*rti.connexdds.SampleInfo property*), 508

coherent_set_sequence_number
(*rti.connexdds.CoherentSetInfo* property), 69

CoherentAccess (class in *rti.connexdds*), 68

CoherentSet (class in *rti.connexdds*), 68

CoherentSetInfo (class in *rti.connexdds*), 68

COLLECTION_TYPE (*rti.connexdds.TypeKind* attribute), 720

COLLECTION_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 722

CollectionType (class in *rti.connexdds*), 69

collector_initial_peers (*rti.connexdds.MonitoringDedicatedParticipantSettings* property), 350

communication (*rti.connexdds.LogCategory* attribute), 343

communication (*rti.connexdds.LogCategory.LogCategory* attribute), 342

compare () (*rti.connexdds.Duration* method), 183

compare () (*rti.connexdds.Time* method), 637

compile () (*rti.connexdds.DynamicData.ContentFilter* method), 184

compile () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilter* method), 370

compile () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilter* method), 426

compile () (*rti.connexdds.ServiceRequest.ContentFilter* method), 524

compile () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilter* method), 584

compile ()
(*rti.connexdds.TopicBuiltinTopicData.ContentFilter* method), 640

compressed_sample_count
(*rti.connexdds.DataReaderCacheStatus* property), 90

compression_ids (*rti.connexdds.CompressionSettings* property), 73

COMPRESSION_LEVEL_BEST_COMPRESSION
(*rti.connexdds.CompressionSettings* attribute), 72

COMPRESSION_LEVEL_BEST_SPEED
(*rti.connexdds.CompressionSettings* attribute), 73

COMPRESSION_LEVEL_DEFAULT
(*rti.connexdds.CompressionSettings* attribute), 73

compression_settings
(*rti.connexdds.DataRepresentation* property), 110

CompressionIdMask (class in *rti.connexdds*), 70

CompressionSettings (class in *rti.connexdds*), 72

compute_crc (*rti.connexdds.WireProtocol* property), 752

concurrency_level
(*rti.connexdds.MonitoringEventDistribution.Settings* property), 351

concurrency_level
(*rti.connexdds.MonitoringLoggingDistributionSettings* property), 352

Condition (class in *rti.connexdds*), 73

condition () (*rti.connexdds.DataReader.Selector* method), 84

condition ()
(*rti.connexdds.DynamicData.DataReader.Selector* method), 188

condition () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector* method), 374

condition () (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector* method), 430

condition ()
(*rti.connexdds.ServiceRequest.DataReader.Selector* method), 527

condition () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector* method), 588

condition () (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector* method), 644

conditions (*rti.connexdds.WaitSet* property), 748

ConditionSeq (class in *rti.connexdds*), 73

CONSTRUCTED_TYPE (*rti.connexdds.TypeKind* attribute), 720

CONSTRUCTED_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 722

contains_entity () (*rti.connexdds.DomainParticipant* method), 158

content () (*rti.connexdds.DataReader.Selector* method), 84

content () (*rti.connexdds.DynamicData.DataReader.Selector* method), 188

content () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector* method), 374

content () (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector* method), 430

content () (*rti.connexdds.ServiceRequest.DataReader.Selector* method), 527

content () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector* method), 588

content () (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector* method), 644

content_filter_allocation
(*rti.connexdds.DomainParticipantResourceLimits* property), 171

content_filter_dropped_sample_count
(*rti.connexdds.DataReaderCacheStatus* property), 90

content_filter_hash_buckets
(*rti.connexdds.DomainParticipantResourceLimits* property), 171

content_filter_property
(*rti.connexdds.SubscriptionBuiltinTopicData* property), 620

content_filter_topic_name
(*rti.connexdds.ContentFilterProperty* property), 76

content_filtered_topic_allocation
(*rti.connexdds.DomainParticipantResourceLimits* property), 171

content_filtered_topic_hash_buckets
(*rti.connexdds.DomainParticipantResourceLimits* property), 171

- content_type (*rti.connexdds.CollectionType* property), 69
- contentfilter_property_max_length (*rti.connexdds.DomainParticipantResourceLimits* property), 171
- ContentFilterBase (class in *rti.connexdds*), 75
- ContentFilteredTopic (class in *rti.connexdds*), 76
- ContentFilteredTopicSeq (class in *rti.connexdds*), 77
- ContentFilterProperty (class in *rti.connexdds*), 76
- CONTINUOUS (*rti.connexdds.TopicQuerySelectionKind* attribute), 688
- CONTINUOUS (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind* attribute), 688
- Cookie (class in *rti.connexdds*), 79
- cookie (*rti.connexdds.AcknowledgmentInfo* property), 35
- cookie (*rti.connexdds.WriteParams* property), 755
- cookie_max_length (*rti.connexdds.DataWriterResourceLimits* property), 133
- CookieSeq (class in *rti.connexdds*), 80
- CookieVector (class in *rti.connexdds*), 82
- corrupted_rtps_message_count (*rti.connexdds.DomainParticipantProtocolStatus* property), 166
- corrupted_rtps_message_count_change (*rti.connexdds.DomainParticipantProtocolStatus* property), 166
- count (*rti.connexdds.ActivityContextMask* property), 39
- count (*rti.connexdds.CompressionIdMask* property), 71
- count (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* property), 152
- count (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* property), 156
- count (*rti.connexdds.InstanceState* property), 312
- count (*rti.connexdds.QosPolicyCount* property), 478
- count (*rti.connexdds.RtpsReservedPortKindMask* property), 502
- count (*rti.connexdds.SampleFlag* property), 506
- count (*rti.connexdds.SampleLostState* property), 512
- count (*rti.connexdds.SampleRejectedState* property), 515
- count (*rti.connexdds.SampleState* property), 518
- count (*rti.connexdds.StatusMask* property), 568
- count (*rti.connexdds.StreamKind* property), 571
- count (*rti.connexdds.ThreadSettingsKindMask* property), 635
- count (*rti.connexdds.TransportBuiltinMask* property), 695
- count (*rti.connexdds.ViewState* property), 746
- count () (*rti.connexdds.AnyDataReaderSeq* method), 44
- count () (*rti.connexdds.AnyDataWriterSeq* method), 48
- count () (*rti.connexdds.AnyTopicSeq* method), 50
- count () (*rti.connexdds.BoolSeq* method), 55
- count () (*rti.connexdds.ChannelSettingsSeq* method), 65
- count () (*rti.connexdds.CharSeq* method), 67
- count () (*rti.connexdds.ConditionSeq* method), 75
- count () (*rti.connexdds.ContentFilteredTopicSeq* method), 79
- count () (*rti.connexdds.CookieSeq* method), 81
- count () (*rti.connexdds.DataReaderSeq* method), 108
- count () (*rti.connexdds.DataWriterSeq* method), 139
- count () (*rti.connexdds.DomainParticipantSeq* method), 177
- count () (*rti.connexdds.DynamicData* method), 222
- count () (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 187
- count () (*rti.connexdds.DynamicData.DataReaderSeq* method), 196
- count () (*rti.connexdds.DynamicData.DataWriterSeq* method), 207
- count () (*rti.connexdds.DynamicDataSeq* method), 251
- count () (*rti.connexdds.DynamicData.TimestampedSeq* method), 253
- count () (*rti.connexdds.DynamicData.TopicSeq* method), 218
- count () (*rti.connexdds.EndpointGroupSeq* method), 258
- count () (*rti.connexdds.EntitySeq* method), 262
- count () (*rti.connexdds.EnumMemberSeq* method), 265
- count () (*rti.connexdds.Float32Seq* method), 276
- count () (*rti.connexdds.Float64Seq* method), 278
- count () (*rti.connexdds.Float128Seq* method), 273
- count () (*rti.connexdds.IAnyDataReaderSeq* method), 293
- count () (*rti.connexdds.IAnyDataWriterSeq* method), 296
- count () (*rti.connexdds.IAnyTopicSeq* method), 298
- count () (*rti.connexdds.IConditionSeq* method), 301
- count () (*rti.connexdds.IEntitySeq* method), 304
- count () (*rti.connexdds.InstanceHandleSeq* method), 310
- count () (*rti.connexdds.Int8Seq* method), 325
- count () (*rti.connexdds.Int16Seq* method), 316
- count () (*rti.connexdds.Int32Seq* method), 318
- count () (*rti.connexdds.Int64Seq* method), 322
- count () (*rti.connexdds.LocatorFilterElementSeq* method), 334
- count () (*rti.connexdds.LocatorSeq* method), 340
- count () (*rti.connexdds.MemberSeq* method), 348
- count () (*rti.connexdds.MonitoringMetricSelectionSeq* method), 355
- count () (*rti.connexdds.MulticastMappingSeq* method), 360
- count () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 373
- count () (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 382
- count () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 394
- count () (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 408
- count () (*rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq* method), 410
- count () (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 403
- count () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 429
- count () (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 439
- count () (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 450
- count () (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 465

- count () (*rti.connexdds.PublicationBuiltinTopicDataTimes-tampedSeq method*), 468
 - count () (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq method*), 459
 - count () (*rti.connexdds.PublisherSeq method*), 477
 - count () (*rti.connexdds.QosPolicyCountSeq method*), 480
 - count () (*rti.connexdds.ServiceRequest.ContentFilteredTopic-Seq method*), 527
 - count () (*rti.connexdds.ServiceRequest.DataReaderSeq method*), 535
 - count () (*rti.connexdds.ServiceRequest.DataWriterSeq method*), 547
 - count () (*rti.connexdds.ServiceRequestSeq method*), 562
 - count () (*rti.connexdds.ServiceRequestTimestampedSeq method*), 564
 - count () (*rti.connexdds.ServiceRequest.TopicSeq method*), 555
 - count () (*rti.connexdds.StringPairSeq method*), 574
 - count () (*rti.connexdds.StringSeq method*), 576
 - count () (*rti.connexdds.SubscriberSeq method*), 584
 - count () (*rti.connexdds.SubscriptionBuiltinTopicData.Content-FilteredTopicSeq method*), 587
 - count () (*rti.connexdds.SubscriptionBuiltinTopic-Data.DataReaderSeq method*), 597
 - count () (*rti.connexdds.SubscriptionBuiltinTopic-Data.DataWriterSeq method*), 609
 - count () (*rti.connexdds.SubscriptionBuiltinTopicDataSeq method*), 624
 - count () (*rti.connexdds.SubscriptionBuiltinTopicDataTimes-tampedSeq method*), 626
 - count () (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq method*), 618
 - count () (*rti.connexdds.TopicBuiltinTopicData.ContentFil-teredTopicSeq method*), 643
 - count () (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq method*), 652
 - count () (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method*), 663
 - count () (*rti.connexdds.TopicBuiltinTopicDataSeq method*), 677
 - count () (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq method*), 680
 - count () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq method*), 672
 - count () (*rti.connexdds.TopicSeq method*), 691
 - count () (*rti.connexdds.TransportInfoSeq method*), 701
 - count () (*rti.connexdds.TransportMulticastSeq method*), 707
 - count () (*rti.connexdds.TransportMulticastSettingsSeq method*), 710
 - count () (*rti.connexdds.TransportUnicastSettingsSeq method*), 715
 - count () (*rti.connexdds.Uint8Seq method*), 736
 - count () (*rti.connexdds.Uint16Seq method*), 728
 - count () (*rti.connexdds.Uint32Seq method*), 730
 - count () (*rti.connexdds.Uint64Seq method*), 733
 - count () (*rti.connexdds.UnionMemberSeq method*), 739
 - count () (*rti.connexdds.WcharSeq method*), 751
 - count () (*rti.connexdds.WstringSeq method*), 758
 - cpu_list (*rti.connexdds.ThreadSettings property*), 631
 - cpu_rotation (*rti.connexdds.ThreadSettings property*), 631
 - create_data () (*rti.connexdds.DynamicData.DataWriter method*), 199
 - create_data () (*rti.connexdds.ParticipantBuiltinTopic-Data.DataWriter method*), 385
 - create_data () (*rti.connexdds.PublicationBuiltinTopic-Data.DataWriter method*), 441
 - create_data () (*rti.connexdds.ServiceRequest.DataWriter method*), 538
 - create_data () (*rti.connexdds.SubscriptionBuiltinTopic-Data.DataWriter method*), 599
 - create_data () (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 654
 - create_from_service_request () (*rti.connexdds.TopicQueryData static method*), 687
 - create_participant_from_config () (*rti.connexdds.QosProvider method*), 481
 - CRITICAL (*rti.connexdds.SyslogVerbosity attribute*), 628
 - critical () (*rti.connexdds.Logger method*), 344
 - current (*rti.connexdds.ProductVersion attribute*), 423
 - current (*rti.connexdds.ProtocolVersion attribute*), 425
 - current_count (*rti.connexdds.PublicationMatchedStatus property*), 469
 - current_count (*rti.connexdds.ServiceRequestAcceptedStatus property*), 558
 - current_count (*rti.connexdds.SubscriptionMatchedStatus property*), 627
 - current_count_change (*rti.connexdds.PublicationMatchedStatus property*), 469
 - current_count_change (*rti.connexdds.SubscriptionMatchedStatus property*), 627
 - current_count_peak (*rti.connexdds.PublicationMatchedStatus property*), 469
 - current_count_peak (*rti.connexdds.SubscriptionMatchedStatus property*), 627
 - current_time (*rti.connexdds.DomainParticipant property*), 158
- D**
- data (*rti.connexdds.DataReader.LoanedSample property*), 83
 - data (*rti.connexdds.DynamicData.LoanedSample property*), 210
 - data (*rti.connexdds.DynamicData.Sample property*), 214
 - data (*rti.connexdds.LoanedDynamicData property*), 331

- data (*rti.connexdds.ParticipantBuiltinTopicData.LoanedSample property*), 395
- data (*rti.connexdds.ParticipantBuiltinTopicData.Sample property*), 399
- data (*rti.connexdds.PublicationBuiltinTopicData.LoanedSample property*), 452
- data (*rti.connexdds.PublicationBuiltinTopicData.Sample property*), 456
- data (*rti.connexdds.ServiceRequest.LoanedSample property*), 548
- data (*rti.connexdds.ServiceRequest.Sample property*), 551
- data (*rti.connexdds.SubscriptionBuiltinTopicData.LoanedSample property*), 610
- data (*rti.connexdds.SubscriptionBuiltinTopicData.Sample property*), 614
- data (*rti.connexdds.TopicBuiltinTopicData.LoanedSample property*), 665
- data (*rti.connexdds.TopicBuiltinTopicData.Sample property*), 668
- DATA_AVAILABLE (*rti.connexdds.StatusMask attribute*), 566
- DATA_ON_READERS (*rti.connexdds.StatusMask attribute*), 566
- data_reader (*rti.connexdds.IReadCondition property*), 305
- data_reader (*rti.connexdds.Query property*), 485
- data_reader_protocol (*rti.connexdds.DataReaderQos property*), 102
- data_reader_resource_limits (*rti.connexdds.DataReaderQos property*), 102
- data_representation (*rti.connexdds.DataReaderQos property*), 102
- data_representation (*rti.connexdds.DataWriterQos property*), 130
- data_representation (*rti.connexdds.TopicQos property*), 685
- data_state (*rti.connexdds.DataStateEx property*), 112
- data_tag (*rti.connexdds.DataReaderQos property*), 102
- data_tag (*rti.connexdds.DataWriterQos property*), 130
- data_tag (*rti.connexdds.PublicationBuiltinTopicData property*), 462
- data_tag (*rti.connexdds.SubscriptionBuiltinTopicData property*), 620
- data_writer_protocol (*rti.connexdds.DataWriterQos property*), 130
- data_writer_resource_limits (*rti.connexdds.DataWriterQos property*), 131
- data_writer_transfer_mode (*rti.connexdds.DataWriterQos property*), 131
- Database (*class in rti.connexdds*), 140
- database (*rti.connexdds.DomainParticipantQos property*), 170
- database (*rti.connexdds.LogCategory attribute*), 343
- database (*rti.connexdds.LogCategory.LogCategory attribute*), 342
- DATABASE_INTEGRATION (*rti.connexdds.ServiceKind attribute*), 521
- DATABASE_INTEGRATION (*rti.connexdds.ServiceKind.ServiceKind attribute*), 522
- DataReader (*class in rti.connexdds*), 83
- datareader (*rti.connexdds.TopicQuery property*), 686
- DATAREADER_CACHE (*rti.connexdds.StatusMask attribute*), 566
- datareader_cache_status (*rti.connexdds.DataReader property*), 87
- datareader_cache_status (*rti.connexdds.DynamicData.DataReader property*), 191
- datareader_cache_status (*rti.connexdds.ParticipantBuiltinTopicData.DataReader property*), 377
- datareader_cache_status (*rti.connexdds.PublicationBuiltinTopicData.DataReader property*), 433
- datareader_cache_status (*rti.connexdds.ServiceRequest.DataReader property*), 530
- datareader_cache_status (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader property*), 591
- datareader_cache_status (*rti.connexdds.TopicBuiltinTopicData.DataReader property*), 647
- DATAREADER_PROTOCOL (*rti.connexdds.StatusMask attribute*), 566
- datareader_protocol_status (*rti.connexdds.DataReader property*), 87
- datareader_protocol_status (*rti.connexdds.DynamicData.DataReader property*), 191
- datareader_protocol_status (*rti.connexdds.ParticipantBuiltinTopicData.DataReader property*), 377
- datareader_protocol_status (*rti.connexdds.PublicationBuiltinTopicData.DataReader property*), 433
- datareader_protocol_status (*rti.connexdds.ServiceRequest.DataReader property*), 530
- datareader_protocol_status (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader property*), 591
- datareader_protocol_status (*rti.connexdds.TopicBuiltinTopicData.DataReader property*), 647
- datareader_qos (*rti.connexdds.QosProvider property*), 482
- datareader_qos_from_profile () (*rti.connexdds.QosProvider method*), 482
- DataReaderCacheStatus (*class in rti.connexdds*), 90
- DataReaderInstanceRemovalKind (*class in rti.connexdds*), 91
- DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind (*class in rti.connexdds*), 91
- DataReaderListener (*class in rti.connexdds*), 94
- DataReader.LoanedSample (*class in rti.connexdds*), 83
- DataReader.LoanedSamples (*class in rti.connexdds*), 83
- DataReaderProtocol (*class in rti.connexdds*), 95
- DataReaderProtocolStatus (*class in rti.connexdds*), 95

DataReaderQos (class in *rti.connextdds*), 97
 DataReaderResourceLimits (class in *rti.connextdds*), 104
 DataReaderResourceLimitsInstanceReplacementSettings (class in *rti.connextdds*), 106
 DataReader.Selector (class in *rti.connextdds*), 84
 DataReaderSeq (class in *rti.connextdds*), 107
 DataRepresentation (class in *rti.connextdds*), 109
 DataState (class in *rti.connextdds*), 110
 DataStateEx (class in *rti.connextdds*), 111
 DataTag (class in *rti.connextdds*), 113
 DataWriter (class in *rti.connextdds*), 114
 DATAWRITER_APPLICATION_ACKNOWLEDGMENT (rti.connextdds.StatusMask attribute), 566
 DATAWRITER_CACHE (rti.connextdds.StatusMask attribute), 566
 datawriter_cache_status (rti.connextdds.DataWriter property), 116
 datawriter_cache_status (rti.connextdds.DynamicData.DataWriter property), 199
 datawriter_cache_status (rti.connextdds.ParticipantBuiltinTopicData.DataWriter property), 385
 datawriter_cache_status (rti.connextdds.PublicationBuiltinTopicData.DataWriter property), 441
 datawriter_cache_status (rti.connextdds.ServiceRequest.DataWriter property), 538
 datawriter_cache_status (rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property), 599
 datawriter_cache_status (rti.connextdds.TopicBuiltinTopicData.DataWriter property), 654
 DATAWRITER_INSTANCE_REPLACED (rti.connextdds.StatusMask attribute), 566
 DATAWRITER_PROTOCOL (rti.connextdds.StatusMask attribute), 566
 datawriter_protocol_status (rti.connextdds.DataWriter property), 116
 datawriter_protocol_status (rti.connextdds.DynamicData.DataWriter property), 199
 datawriter_protocol_status (rti.connextdds.ParticipantBuiltinTopicData.DataWriter property), 385
 datawriter_protocol_status (rti.connextdds.PublicationBuiltinTopicData.DataWriter property), 441
 datawriter_protocol_status (rti.connextdds.ServiceRequest.DataWriter property), 538
 datawriter_protocol_status (rti.connextdds.SubscriptionBuiltinTopicData.DataWriter property), 600
 datawriter_protocol_status (rti.connextdds.TopicBuiltinTopicData.DataWriter property), 655
 datawriter_qos (rti.connextdds.QosProvider property), 482
 datawriter_qos_from_profile () (rti.connextdds.QosProvider method), 482
 datawriter_qos_profile_name (rti.connextdds.MonitoringEventDistributionSettings property), 351
 datawriter_qos_profile_name (rti.connextdds.MonitoringLoggingDistributionSettings property), 352
 datawriter_qos_profile_name (rti.connextdds.MonitoringPeriodicDistributionSettings property), 356
 DataWriterCacheStatus (class in *rti.connextdds*), 120
 DataWriterListener (class in *rti.connextdds*), 121
 DataWriterProtocol (class in *rti.connextdds*), 122
 DataWriterProtocolStatus (class in *rti.connextdds*), 122
 DataWriterQos (class in *rti.connextdds*), 124
 DataWriterResourceLimits (class in *rti.connextdds*), 132
 DataWriterResourceLimitsInstanceReplacementKind (class in *rti.connextdds*), 134
 DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind (class in *rti.connextdds*), 134
 DataWriterSeq (class in *rti.connextdds*), 137
 DataWriterShmemRefTransferModeSettings (class in *rti.connextdds*), 139
 DataWriterTransferMode (class in *rti.connextdds*), 140
 dds_built_in_endpoints (rti.connextdds.ParticipantBuiltinTopicData property), 405
 Deadline (class in *rti.connextdds*), 141
 deadline (rti.connextdds.DataReaderQos property), 102
 deadline (rti.connextdds.DataWriterQos property), 131
 deadline (rti.connextdds.PublicationBuiltinTopicData property), 462
 deadline (rti.connextdds.SubscriptionBuiltinTopicData property), 620
 deadline (rti.connextdds.TopicBuiltinTopicData property), 675
 deadline (rti.connextdds.TopicQos property), 685
 DEBUG (rti.connextdds.PrintFormat attribute), 418
 DEBUG (rti.connextdds.PrintFormat.PrintFormat attribute), 419
 DEBUG (rti.connextdds.SyslogVerbosity attribute), 628
 DEBUG (rti.logging.distlog.LogLevel attribute), 770
 debug () (rti.connextdds.Logger method), 344
 debug () (rti.logging.distlog.Logger static method), 771
 dedicated_participant (rti.connextdds.MonitoringDistributionSettings property), 350
 DEFAULT (rti.connextdds.ActivityContextMask attribute), 37
 DEFAULT (rti.connextdds.DynamicDataEncapsulationKind attribute), 245
 DEFAULT (rti.connextdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind attribute),

- 245
- DEFAULT (*rti.connexdds.DynamicDataTypeSerializationProperty attribute*), 254
- DEFAULT (*rti.connexdds.PrintFormat attribute*), 418
- DEFAULT (*rti.connexdds.PrintFormatKind attribute*), 420
- DEFAULT (*rti.connexdds.PrintFormatKind.PrintFormatKind attribute*), 421
- DEFAULT (*rti.connexdds.PrintFormat.PrintFormat attribute*), 419
- default (*rti.connexdds.PrintFormatProperty attribute*), 422
- default (*rti.connexdds.QosProvider attribute*), 482
- default_dataareader_qos (*rti.connexdds.DomainParticipant property*), 158
- default_dataareader_qos (*rti.connexdds.Subscriber property*), 579
- default_datawriter_qos (*rti.connexdds.DomainParticipant property*), 158
- default_datawriter_qos (*rti.connexdds.Publisher property*), 473
- default_domain_announcement_period (*rti.connexdds.DiscoveryConfig property*), 148
- default_filter_state (*rti.connexdds.DataReader property*), 87
- default_filter_state (*rti.connexdds.DynamicData.DataReader property*), 191
- default_filter_state (*rti.connexdds.ParticipantBuiltinTopicData.DataReader property*), 377
- default_filter_state (*rti.connexdds.PublicationBuiltinTopicData.DataReader property*), 433
- default_filter_state (*rti.connexdds.ServiceRequest.DataReader property*), 530
- default_filter_state (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader property*), 592
- default_filter_state (*rti.connexdds.TopicBuiltinTopicData.DataReader property*), 647
- DEFAULT_LABEL (*rti.connexdds.UnionMember attribute*), 737
- default_library (*rti.connexdds.QosProvider property*), 482
- DEFAULT_MASK (*rti.connexdds.RtpsReservedPortKindMask attribute*), 500
- DEFAULT_NAME (*rti.connexdds.FlowController attribute*), 279
- default_participant_qos (*rti.connexdds.DomainParticipant attribute*), 158
- default_profile (*rti.connexdds.QosProvider property*), 482
- default_profile_library (*rti.connexdds.QosProvider property*), 482
- default_provider_params (*rti.connexdds.QosProvider attribute*), 482
- DEFAULT_PUBLICATION (*rti.connexdds.CompressionIdMask attribute*), 70
- default_publisher_qos (*rti.connexdds.DomainParticipant property*), 158
- default_subscriber_qos (*rti.connexdds.DomainParticipant property*), 159
- DEFAULT_SUBSCRIPTION (*rti.connexdds.CompressionIdMask attribute*), 70
- default_topic_qos (*rti.connexdds.DomainParticipant property*), 159
- default_unicast (*rti.connexdds.DomainParticipantQos property*), 170
- default_unicast_locators (*rti.connexdds.ParticipantBuiltinTopicData property*), 405
- DELETE (*rti.connexdds.PersistentJournalKind attribute*), 412
- delete_durable_subscription() (*rti.connexdds.DomainParticipant method*), 159
- depth (*rti.connexdds.History property*), 288
- deserialized_type_object_dynamic_allocation_threshold (*rti.connexdds.DomainParticipantResourceLimits property*), 172
- destination_order (*rti.connexdds.DataReaderQos property*), 102
- destination_order (*rti.connexdds.DataWriterQos property*), 131
- destination_order (*rti.connexdds.PublicationBuiltinTopicData property*), 462
- destination_order (*rti.connexdds.SubscriptionBuiltinTopicData property*), 620
- destination_order (*rti.connexdds.TopicBuiltinTopicData property*), 675
- destination_order (*rti.connexdds.TopicQos property*), 685
- DESTINATION_UNREACHABLE (*rti.connexdds.StatusMask attribute*), 566
- DestinationOrder (*class in rti.connexdds*), 142
- DestinationOrderKind (*class in rti.connexdds*), 142
- DestinationOrderKind.DestinationOrderKind (*class in rti.connexdds*), 142
- DestinationOrderScopeKind (*class in rti.connexdds*), 144
- DestinationOrderScopeKind.DestinationOrderScopeKind (*class in rti.connexdds*), 144
- detach_all() (*rti.connexdds.WaitSet method*), 748
- detach_condition() (*rti.connexdds.WaitSet method*), 748
- detached_instance_count (*rti.connexdds.DataReaderCacheStatus property*), 90
- detached_instance_count_peak (*rti.connexdds.DataReaderCacheStatus property*), 90
- dimension() (*rti.connexdds.ArrayType method*), 51
- dimension_count (*rti.connexdds.ArrayType property*), 51
- direct_communication (*rti.connexdds.Durability property*), 178
- disable_asynchronous_batch (*rti.connexdds.AsynchronousPublisher property*), 52
- disable_asynchronous_write (*rti.connexdds.AsynchronousPublisher property*), 52

disable_fragmentation_support
 (*rti.connexdds.BuiltinTopicReaderResourceLimits*
 property), 59
 disable_fragmentation_support
 (*rti.connexdds.DataReaderResourceLimits* *property*),
 104
 disable_inline_keyhash
 (*rti.connexdds.DataWriterProtocol* *property*), 122
 disable_positive_acks
 (*rti.connexdds.DataReaderProtocol* *property*), 95
 disable_positive_acks
 (*rti.connexdds.DataWriterProtocol* *property*), 122
 disable_positive_acks
 (*rti.connexdds.PublicationBuiltinTopicData* *property*),
 462
 disable_positive_acks
 (*rti.connexdds.SubscriptionBuiltinTopicData*
 property), 621
 disable_positive_acks_decrease_sample_keep_duration_factor
 (*rti.connexdds.RtpsReliableWriterProtocol* *property*),
 498
 disable_positive_acks_enable_adaptive_sample_keep_duration
 (*rti.connexdds.RtpsReliableWriterProtocol* *property*),
 498
 disable_positive_acks_increase_sample_keep_duration_factor
 (*rti.connexdds.RtpsReliableWriterProtocol* *property*),
 498
 disable_positive_acks_max_sample_keep_duration
 (*rti.connexdds.RtpsReliableWriterProtocol*
 property), 498
 disable_positive_acks_min_sample_keep_duration
 (*rti.connexdds.RtpsReliableWriterProtocol*
 property), 498
 disable_repair_piggyback_heartbeat
 (*rti.connexdds.RtpsReliableWriterProtocol* *property*),
 499
 disable_topic_query_publication
 (*rti.connexdds.AsynchronousPublisher* *property*), 52
 disabled (*rti.connexdds.AsynchronousPublisher* *attribute*), 52
 disabled (*rti.connexdds.Batch* *attribute*), 53
 disabled (*rti.connexdds.Monitoring* *attribute*), 349
 disabled_metrics_selection
 (*rti.connexdds.MonitoringMetricSelection* *property*),
 353
 disallow_type_coercion
 (*rti.connexdds.TypeConsistencyEnforcement*
 attribute), 717
 DISALLOW_TYPE_COERCION
 (*rti.connexdds.TypeConsistencyEnforcementKind*
 attribute), 717
 DISALLOW_TYPE_COERCION (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind*
 attribute), 718
 discovered_participant_data()
 (*rti.connexdds.DomainParticipant* *method*), 159
 discovered_participant_subject_name()
 (*rti.connexdds.DomainParticipant* *method*), 159
 discovered_participants()
 (*rti.connexdds.DomainParticipant* *method*), 159
 discovered_participants_from_subject_name()
 (*rti.connexdds.DomainParticipant*
 method), 159
 discovered_topic_data()
 (*rti.connexdds.DomainParticipant* *method*), 159
 discovered_topics()
 (*rti.connexdds.DomainParticipant*
 method), 160
 Discovery (*class in rti.connexdds*), 147
 discovery (*rti.connexdds.DomainParticipantQos* *property*),
 170
 discovery (*rti.connexdds.LogCategory* *attribute*), 343
 discovery (*rti.connexdds.LogCategory.LogCategory*
 attribute), 342
 discovery_config (*rti.connexdds.DomainParticipantQos*
 property), 170
 DiscoveryConfig (*class in rti.connexdds*), 147
 DiscoveryConfigBuiltinChannelKindMask (*class*
 in rti.connexdds), 150
 DiscoveryConfigBuiltinPluginKindMask (*class in*
 rti.connexdds), 153
 discriminator (*rti.connexdds.UnionType* *property*), 740
 discriminator_value (*rti.connexdds.DynamicData*
 property), 222
 dispatch()
 (*rti.connexdds.ICondition* *method*), 299
 dispatch()
 (*rti.connexdds.StatusCondition* *method*), 565
 dispatch()
 (*rti.connexdds.WaitSet* *method*), 748
 dispatch_async()
 (*rti.connexdds.WaitSet* *method*), 748
 dispose_instance()
 (*rti.connexdds.DataWriter* *method*),
 116
 dispose_instance()
 (*rti.connexdds.DynamicData.DataWriter* *method*),
 199
 dispose_instance()
 (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter*
 method),
 385
 dispose_instance()
 (*rti.connexdds.PublicationBuiltinTopicData.DataWriter*
 method),
 441
 dispose_instance()
 (*rti.connexdds.ServiceRequest.DataWriter* *method*),
 538
 dispose_instance()
 (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter*
 method),
 600
 dispose_instance()
 (*rti.connexdds.TopicBuiltinTopicData.DataWriter*
 method), 655
 dispose_instance_async()
 (*rti.connexdds.DynamicData.DataWriter* *method*),
 199
 dispose_instance_async()
 (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter*
 method),
 385
 dispose_instance_async()
 (*rti.connexdds.PublicationBuiltinTopicData.DataWriter*
 method),
 441

- dispose_instance_async () (rti.connexdds.ServiceRequest.DataWriter method), 538
- dispose_instance_async () (rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method), 600
- dispose_instance_async () (rti.connexdds.TopicBuiltinTopicData.DataWriter method), 655
- DISPOSED (rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind attribute), 134
- DISPOSED (rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute), 135
- disposed (rti.connexdds.GenerationCount property), 285
- disposed_instance_count (rti.connexdds.DataReaderCacheStatus property), 90
- disposed_instance_count (rti.connexdds.DataWriterCacheStatus property), 120
- disposed_instance_count_peak (rti.connexdds.DataReaderCacheStatus property), 90
- disposed_instance_count_peak (rti.connexdds.DataWriterCacheStatus property), 120
- disposed_instance_removal (rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings property), 107
- DISPOSED_THEN_ALIVE (rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind attribute), 134
- DISPOSED_THEN_ALIVE (rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind attribute), 135
- DistlogHandler (class in rti.logging.handler), 773
- distribution_settings (rti.connexdds.Monitoring property), 349
- dll (rti.connexdds.TransportMulticastMappingFunction property), 706
- dns_tracker_polling_period (rti.connexdds.DiscoveryConfig property), 148
- domain_entity_qos_library_name (rti.connexdds.DomainParticipantConfigParams property), 164
- domain_entity_qos_profile_name (rti.connexdds.DomainParticipantConfigParams property), 164
- DOMAIN_ID (rti.connexdds.ActivityContextMask attribute), 37
- domain_id (rti.connexdds.DomainParticipant property), 160
- domain_id (rti.connexdds.DomainParticipantConfigParams property), 164
- domain_id (rti.connexdds.MonitoringDedicatedParticipantSettings property), 350
- domain_id (rti.connexdds.ParticipantBuiltinTopicData property), 405
- domain_id (rti.logging.distlog.LoggerOptions property), 772
- domain_id_gain (rti.connexdds.RtpsWellKnownPorts property), 504
- DOMAIN_ID_USE_XML_CONFIG (rti.connexdds.DomainParticipantConfigParams attribute), 163
- DomainParticipant (class in rti.connexdds), 157
- DomainParticipantConfigParams (class in rti.connexdds), 163
- DomainParticipantFactoryQos (class in rti.connexdds), 164
- DomainParticipantListener (class in rti.connexdds), 166
- DomainParticipantProtocolStatus (class in rti.connexdds), 166
- DomainParticipantQos (class in rti.connexdds), 166
- DomainParticipantResourceLimits (class in rti.connexdds), 171
- DomainParticipantSeq (class in rti.connexdds), 175
- DoubleSeq (in module rti.connexdds), 178
- DoubleType (in module rti.connexdds), 178
- drop_incomplete_coherent_set (rti.connexdds.Presentation property), 415
- dropped_fragment_count (rti.connexdds.DataReaderProtocolStatus property), 96
- duplicate_sample_bytes (rti.connexdds.DataReaderProtocolStatus property), 96
- duplicate_sample_count (rti.connexdds.DataReaderProtocolStatus property), 96
- Durability (class in rti.connexdds), 178
- durability (rti.connexdds.DataReaderQos property), 102
- durability (rti.connexdds.DataWriterQos property), 131
- durability (rti.connexdds.PublicationBuiltinTopicData property), 462
- durability (rti.connexdds.SubscriptionBuiltinTopicData property), 621
- durability (rti.connexdds.TopicBuiltinTopicData property), 675
- durability (rti.connexdds.TopicQos property), 685
- durability_service (rti.connexdds.DataWriterQos property), 131
- durability_service (rti.connexdds.PublicationBuiltinTopicData property), 462
- durability_service (rti.connexdds.TopicBuiltinTopicData property), 675
- durability_service (rti.connexdds.TopicQos property), 685
- DurabilityKind (class in rti.connexdds), 179
- DurabilityKind.DurabilityKind (class in rti.connexdds), 179
- DurabilityService (class in rti.connexdds), 181
- Duration (class in rti.connexdds), 182
- duration (rti.connexdds.LatencyBudget property), 326
- duration (rti.connexdds.Lifespan property), 327
- dynamically_allocate_fragmented_samples (rti.connexdds.BuiltinTopicReaderResourceLimits property), 59

dynamically_allocate_fragmented_samples
(*rti.connexdds.DataReaderResourceLimits* property), 104

DynamicData (class in *rti.connexdds*), 184

DynamicData.ContentFilter (class in *rti.connexdds*), 184

DynamicData.ContentFilteredTopic (class in *rti.connexdds*), 185

DynamicData.ContentFilteredTopicSeq (class in *rti.connexdds*), 186

DynamicData.DataReader (class in *rti.connexdds*), 188

DynamicData.DataReaderListener (class in *rti.connexdds*), 194

DynamicData.DataReader.Selector (class in *rti.connexdds*), 188

DynamicData.DataReaderSeq (class in *rti.connexdds*), 195

DynamicData.DataWriter (class in *rti.connexdds*), 197

DynamicData.DataWriterListener (class in *rti.connexdds*), 205

DynamicData.DataWriterSeq (class in *rti.connexdds*), 206

DynamicDataEncapsulationKind (class in *rti.connexdds*), 245

DynamicDataEncapsulationKind.Dynamic-DataEncapsulationKind (class in *rti.connexdds*), 245

DynamicData.FieldsIterator (class in *rti.connexdds*), 208

DynamicData.FieldsView (class in *rti.connexdds*), 208

DynamicData.IndexIterator (class in *rti.connexdds*), 209

DynamicDataInfo (class in *rti.connexdds*), 247

DynamicData.ItemsIterator (class in *rti.connexdds*), 209

DynamicData.ItemsView (class in *rti.connexdds*), 209

DynamicData.ITopicDescription (class in *rti.connexdds*), 209

DynamicData.LoanedSample (class in *rti.connexdds*), 210

DynamicData.LoanedSamples (class in *rti.connexdds*), 210

DynamicDataMemberInfo (class in *rti.connexdds*), 248

DynamicData.NoOpDataReaderListener (class in *rti.connexdds*), 211

DynamicData.NoOpDataWriterListener (class in *rti.connexdds*), 212

DynamicData.NoOpTopicListener (class in *rti.connexdds*), 213

DynamicDataProperty (class in *rti.connexdds*), 249

DynamicData.Sample (class in *rti.connexdds*), 213

DynamicDataSeq (class in *rti.connexdds*), 249

DynamicData.SharedSamples (class in *rti.connexdds*), 214

DynamicData.TimestampedSeq (class in *rti.connexdds*), 251

DynamicData.Topic (class in *rti.connexdds*), 214

DynamicData.TopicDescription (class in *rti.connexdds*), 215

DynamicData.TopicListener (class in *rti.connexdds*), 216

DynamicData.TopicSeq (class in *rti.connexdds*), 216

DynamicData.TopicTypeSupport (class in *rti.connexdds*), 218

DynamicDataTypeSerializationProperty (class in *rti.connexdds*), 254

DynamicData.ValidLoanedSamples (class in *rti.connexdds*), 219

DynamicData.WriterContentFilter (class in *rti.connexdds*), 219

DynamicData.WriterContentFilterHelper (class in *rti.connexdds*), 220

DynamicType (class in *rti.connexdds*), 255

DynamicTypePrintFormatProperty (class in *rti.connexdds*), 255

E

EARLIEST_DEADLINE_FIRST
(*rti.connexdds.FlowControllerSchedulingPolicy* attribute), 281

EARLIEST_DEADLINE_FIRST
(*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy* attribute), 282

echo_to_stdout (*rti.logging.distlog.LoggerOptions* property), 772

element_count (*rti.connexdds.DynamicDataMemberInfo* property), 248

element_kind (*rti.connexdds.DynamicDataMemberInfo* property), 248

EMERGENCY (*rti.connexdds.Syslog* Verbosity attribute), 628

emergency () (*rti.connexdds.Logger* method), 344

emit () (*rti.logging.handler.DistlogHandler* method), 773

EMPTY_INSTANCES
(*rti.connexdds.DataReaderInstanceRemovalKind* attribute), 93

EMPTY_INSTANCES (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind* attribute), 92

empty_reliable_writer_cache
(*rti.connexdds.ReliableWriterCacheChangedStatus* property), 492

enable (*rti.connexdds.Batch* property), 53

enable (*rti.connexdds.Monitoring* property), 349

enable (*rti.connexdds.MonitoringDedicatedParticipantSettings* property), 350

enable (*rti.connexdds.PersistentStorageSettings* property), 413

enable (*rti.connexdds.TopicQueryDispatch* property), 687

enable () (*rti.connexdds.IEntity* method), 302

enable_data_consistency_check (*rti.connexdds.DataWriterShmemRefTransferModeSettings* property), 140

enable_endpoint_discovery (*rti.connexdds.Discovery* property), 147

enable_multicast_periodic_heartbeat
(*rti.connexdds.RtpsReliableWriterProtocol* property), 499

- enable_required_subscriptions (*rti.connexdds.Availability property*), 52
- enabled (*rti.connexdds.Batch attribute*), 53
- enabled (*rti.connexdds.IEntity property*), 302
- enabled (*rti.connexdds.Monitoring attribute*), 349
- enabled() (*rti.connexdds.AsynchronousPublisher static method*), 52
- enabled_builtin_channels (*rti.connexdds.DiscoveryConfig property*), 148
- enabled_metrics_selection (*rti.connexdds.MonitoringMetricSelection property*), 353
- enabled_statuses (*rti.connexdds.StatusCondition property*), 565
- enabled_transports (*rti.connexdds.Discovery property*), 147
- enabled_with_max_data_bytes() (*rti.connexdds.Batch static method*), 53
- enabled_with_max_samples() (*rti.connexdds.Batch static method*), 53
- encapsulation_id (*rti.connexdds.SampleInfo property*), 508
- encapsulation_kind (*rti.connexdds.DynamicDataInfo property*), 248
- end() (*rti.connexdds.CoherentAccess method*), 68
- end() (*rti.connexdds.CoherentSet method*), 68
- endpoint_type_object_lb_serialization_threshold (*rti.connexdds.DiscoveryConfig property*), 148
- EndpointGroup (*class in rti.connexdds*), 256
- EndpointGroupSeq (*class in rti.connexdds*), 256
- EndpointGroupVector (*class in rti.connexdds*), 258
- entities (*rti.connexdds.LogCategory attribute*), 344
- entities (*rti.connexdds.LogCategory.LogCategory attribute*), 342
- Entity (*class in rti.connexdds*), 259
- entity (*rti.connexdds.StatusCondition property*), 565
- entity_factory (*rti.connexdds.DomainParticipantFactoryQos property*), 165
- entity_factory (*rti.connexdds.DomainParticipantQos property*), 170
- entity_factory (*rti.connexdds.PublisherQos property*), 475
- entity_factory (*rti.connexdds.SubscriberQos property*), 582
- ENTITY_KIND (*rti.connexdds.ActivityContextMask attribute*), 38
- ENTITY_NAME (*rti.connexdds.ActivityContextMask attribute*), 38
- entity_name (*rti.connexdds.DataReaderQos property*), 102
- entity_name (*rti.connexdds.DataWriterQos property*), 131
- entity_name (*rti.connexdds.PublisherQos property*), 475
- entity_name (*rti.connexdds.SubscriberQos property*), 582
- ENTITY_NAME_USE_XML_CONFIG (*rti.connexdds.DomainParticipantConfigParams attribute*), 163
- EntityFactory (*class in rti.connexdds*), 260
- EntityName (*class in rti.connexdds*), 260
- EntitySeq (*class in rti.connexdds*), 261
- enum_as_int (*rti.connexdds.PrintFormatProperty property*), 422
- ENUMERATION_TYPE (*rti.connexdds.TypeKind attribute*), 720
- ENUMERATION_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722
- EnumMember (*class in rti.connexdds*), 263
- EnumMemberSeq (*class in rti.connexdds*), 263
- EnumType (*class in rti.connexdds*), 265
- Error, 266
- ERROR (*rti.connexdds.SyslogVerbosity attribute*), 629
- ERROR (*rti.logging.distlog.LogLevel attribute*), 771
- error() (*rti.connexdds.Logger method*), 344
- error() (*rti.logging.distlog.Logger static method*), 771
- evaluate() (*rti.connexdds.DynamicData.ContentFilter method*), 184
- evaluate() (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilter method*), 370
- evaluate() (*rti.connexdds.PublicationBuiltinTopicData.ContentFilter method*), 426
- evaluate() (*rti.connexdds.ServiceRequest.ContentFilter method*), 524
- evaluate() (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilter method*), 584
- evaluate() (*rti.connexdds.TopicBuiltinTopicData.ContentFilter method*), 640
- Event (*class in rti.connexdds*), 266
- event (*rti.connexdds.DomainParticipantQos property*), 170
- event_count (*rti.connexdds.WaitSetProperty property*), 749
- event_delay (*rti.connexdds.WaitSetProperty property*), 749
- event_settings (*rti.connexdds.MonitoringDistributionSettings property*), 350
- EventCount32 (*class in rti.connexdds*), 267
- EventCount64 (*class in rti.connexdds*), 267
- Exception, 267
- EXCEPTION (*rti.connexdds.Verbosity attribute*), 742
- EXCEPTION (*rti.connexdds.Verbosity.Verbosity attribute*), 743
- exclusive (*rti.connexdds.Ownership attribute*), 367
- EXCLUSIVE (*rti.connexdds.OwnershipKind attribute*), 367
- EXCLUSIVE (*rti.connexdds.OwnershipKind.OwnershipKind attribute*), 367
- exclusive_area (*rti.connexdds.PublisherQos property*), 476
- exclusive_area (*rti.connexdds.SubscriberQos property*), 582
- exclusive_ea (*rti.connexdds.ExclusiveArea attribute*), 268
- ExclusiveArea (*class in rti.connexdds*), 268
- exists() (*rti.connexdds.DataTag method*), 113
- exists() (*rti.connexdds.Property method*), 424
- expects_inline_qos (*rti.connexdds.DataReaderProtocol property*), 95
- expiration_time (*rti.connexdds.InvalidLocalIdentityAdvanceNoticeStatus property*), 326
- expired_dropped_sample_count (*rti.connexdds.DataReaderCacheStatus property*), 90

- expression (*rti.connexdds.Filter* property), 271
- expression (*rti.connexdds.Query* property), 485
- expression (*rti.connexdds.QueryCondition* property), 486
- ExpressionProperty (class in *rti.connexdds*), 268
- extend () (*rti.connexdds.AnyDataReaderSeq* method), 44
- extend () (*rti.connexdds.AnyDataWriterSeq* method), 48
- extend () (*rti.connexdds.AnyTopicSeq* method), 50
- extend () (*rti.connexdds.BoolSeq* method), 55
- extend () (*rti.connexdds.ChannelSettingsSeq* method), 65
- extend () (*rti.connexdds.CharSeq* method), 67
- extend () (*rti.connexdds.ConditionSeq* method), 75
- extend () (*rti.connexdds.ContentFilteredTopicSeq* method), 79
- extend () (*rti.connexdds.CookieSeq* method), 81
- extend () (*rti.connexdds.DataReaderSeq* method), 108
- extend () (*rti.connexdds.DataWriterSeq* method), 139
- extend () (*rti.connexdds.DomainParticipantSeq* method), 177
- extend () (*rti.connexdds.DynamicData* method), 222
- extend ()
 - (*rti.connexdds.DynamicData.ContentFilteredTopicSeq* method), 187
- extend () (*rti.connexdds.DynamicData.DataReaderSeq* method), 196
- extend () (*rti.connexdds.DynamicData.DataWriterSeq* method), 208
- extend () (*rti.connexdds.DynamicDataSeq* method), 251
- extend () (*rti.connexdds.DynamicDataTimestampedSeq* method), 253
- extend () (*rti.connexdds.DynamicData.TopicSeq* method), 218
- extend () (*rti.connexdds.EndpointGroupSeq* method), 258
- extend () (*rti.connexdds.EntitySeq* method), 262
- extend () (*rti.connexdds.EnumMemberSeq* method), 265
- extend () (*rti.connexdds.Float32Seq* method), 276
- extend () (*rti.connexdds.Float64Seq* method), 278
- extend () (*rti.connexdds.Float128Seq* method), 273
- extend () (*rti.connexdds.IAnyDataReaderSeq* method), 293
- extend () (*rti.connexdds.IAnyDataWriterSeq* method), 296
- extend () (*rti.connexdds.IAnyTopicSeq* method), 298
- extend () (*rti.connexdds.IConditionSeq* method), 301
- extend () (*rti.connexdds.IEntitySeq* method), 304
- extend () (*rti.connexdds.InstanceHandleSeq* method), 310
- extend () (*rti.connexdds.Int8Seq* method), 325
- extend () (*rti.connexdds.Int16Seq* method), 316
- extend () (*rti.connexdds.Int32Seq* method), 318
- extend () (*rti.connexdds.Int64Seq* method), 322
- extend () (*rti.connexdds.LocatorFilterElementSeq* method), 334
- extend () (*rti.connexdds.LocatorSeq* method), 340
- extend () (*rti.connexdds.MemberSeq* method), 348
- extend () (*rti.connexdds.MonitoringMetricSelectionSeq* method), 355
- extend () (*rti.connexdds.MulticastMappingSeq* method), 360
- extend () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq* method), 373
- extend () (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq* method), 382
- extend () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq* method), 394
- extend () (*rti.connexdds.ParticipantBuiltinTopicDataSeq* method), 408
- extend () (*rti.connexdds.ParticipantBuiltinTopicData.TimestampedSeq* method), 410
- extend () (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq* method), 403
- extend () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq* method), 429
- extend () (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq* method), 439
- extend () (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq* method), 450
- extend () (*rti.connexdds.PublicationBuiltinTopicDataSeq* method), 465
- extend () (*rti.connexdds.PublicationBuiltinTopicData.TimestampedSeq* method), 468
- extend ()
 - (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq* method), 459
- extend () (*rti.connexdds.PublisherSeq* method), 477
- extend () (*rti.connexdds.QosPolicyCountSeq* method), 480
- extend () (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq* method), 527
- extend () (*rti.connexdds.ServiceRequest.DataReaderSeq* method), 535
- extend () (*rti.connexdds.ServiceRequest.DataWriterSeq* method), 547
- extend () (*rti.connexdds.ServiceRequestSeq* method), 562
- extend () (*rti.connexdds.ServiceRequestTimestampedSeq* method), 564
- extend () (*rti.connexdds.ServiceRequest.TopicSeq* method), 555
- extend () (*rti.connexdds.StringPairSeq* method), 574
- extend () (*rti.connexdds.StringSeq* method), 576
- extend () (*rti.connexdds.SubscriberSeq* method), 584
- extend () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq* method), 588
- extend () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq* method), 597
- extend () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq* method), 609
- extend () (*rti.connexdds.SubscriptionBuiltinTopicDataSeq* method), 624
- extend () (*rti.connexdds.SubscriptionBuiltinTopicData.TimestampedSeq* method), 627
- extend ()

(rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq method), 618
 extend() (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 643
 extend() (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq method*), 652
 extend() (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method*), 664
 extend() (*rti.connexdds.TopicBuiltinTopicDataSeq method*), 677
 extend() (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq method*), 680
 extend() (*rti.connexdds.TopicBuiltinTopicData.TopicSeq method*), 672
 extend() (*rti.connexdds.TopicSeq method*), 692
 extend() (*rti.connexdds.TransportInfoSeq method*), 701
 extend() (*rti.connexdds.TransportMulticastSeq method*), 707
 extend() (*rti.connexdds.TransportMulticastSettingsSeq method*), 710
 extend() (*rti.connexdds.TransportUnicastSettingsSeq method*), 715
 extend() (*rti.connexdds.Uint8Seq method*), 736
 extend() (*rti.connexdds.Uint16Seq method*), 728
 extend() (*rti.connexdds.Uint32Seq method*), 730
 extend() (*rti.connexdds.Uint64Seq method*), 733
 extend() (*rti.connexdds.UnionMemberSeq method*), 739
 extend() (*rti.connexdds.WcharSeq method*), 751
 extend() (*rti.connexdds.WstringSeq method*), 758
 extensibility_kind (*rti.connexdds.EnumType property*), 266
 extensibility_kind (*rti.connexdds.StructType property*), 578
 extensibility_kind (*rti.connexdds.UnionType property*), 741
 extensibility_kind() (*rti.connexdds.ACTEnumMember method*), 32
 extensibility_kind() (*rti.connexdds.ACTMember method*), 33
 extensibility_kind() (*rti.connexdds.ACTUnionMember method*), 34
 ExtensibilityKind (*class in rti.connexdds*), 269
 ExtensibilityKind.ExtensibilityKind (*class in rti.connexdds*), 269
 EXTENSIBLE (*rti.connexdds.ExtensibilityKind attribute*), 269
 EXTENSIBLE (*rti.connexdds.ExtensibilityKind.ExtensibilityKind attribute*), 269

F

fast_heartbeat_period (*rti.connexdds.RtpsReliableWriterProtocol property*), 499
 FATAL (*rti.logging.distlog.LogLevel attribute*), 771
 fatal() (*rti.logging.distlog.Logger static method*), 771
 fields() (*rti.connexdds.DynamicData method*), 222
 file_name (*rti.connexdds.PersistentStorageSettings property*), 413
 Filter (*class in rti.connexdds*), 271
 filter (*rti.connexdds.TopicQuerySelection property*), 688
 filter_class_name (*rti.connexdds.ContentFilterProperty property*), 76
 filter_expression (*rti.connexdds.ChannelSettings property*), 63
 filter_expression (*rti.connexdds.ContentFilteredTopic property*), 77
 filter_expression (*rti.connexdds.ContentFilterProperty property*), 76
 filter_expression (*rti.connexdds.DynamicData.ContentFilteredTopic property*), 185
 filter_expression (*rti.connexdds.LocatorFilterElement property*), 333
 filter_expression (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic property*), 371
 filter_expression (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic property*), 427
 filter_expression (*rti.connexdds.ServiceRequest.ContentFilteredTopic property*), 524
 filter_expression (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic property*), 585
 filter_expression (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic property*), 641
 filter_level (*rti.logging.distlog.LoggerOptions property*), 772
 filter_level() (*rti.logging.distlog.Logger static method*), 771
 filter_name (*rti.connexdds.LocatorFilter property*), 332
 filter_name (*rti.connexdds.MultiChannel property*), 357
 filter_parameters (*rti.connexdds.ContentFilteredTopic property*), 77
 filter_parameters (*rti.connexdds.DynamicData.ContentFilteredTopic property*), 185
 filter_parameters (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic property*), 371
 filter_parameters (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic property*), 427
 filter_parameters (*rti.connexdds.ServiceRequest.ContentFilteredTopic property*), 525
 filter_parameters (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic property*), 585
 filter_parameters (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic property*), 641
 FilterSampleInfo (*class in rti.connexdds*), 271

- `FINAL` (*rti.connexdds.ExtensibilityKind* attribute), 270
- `FINAL` (*rti.connexdds.ExtensibilityKind.ExtensibilityKind* attribute), 269
- `finalize()` (*rti.connexdds.DynamicData.ContentFilter* method), 185
- `finalize()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilter* method), 370
- `finalize()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilter* method), 426
- `finalize()` (*rti.connexdds.ServiceRequest.ContentFilter* method), 524
- `finalize()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilter* method), 585
- `finalize()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilter* method), 640
- `finalize()` (*rti.logging.distlog.Logger* static method), 771
- `finalize_participant_factory()` (*rti.connexdds.DomainParticipant* static method), 160
- `find()` (*rti.connexdds.ContentFilteredTopic* static method), 77
- `find()` (*rti.connexdds.DomainParticipant* static method), 160
- `find()` (*rti.connexdds.DynamicData.ContentFilteredTopic* static method), 185
- `find()` (*rti.connexdds.DynamicData.Topic* static method), 215
- `find()` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic* static method), 371
- `find()` (*rti.connexdds.ParticipantBuiltinTopicData.Topic* static method), 400
- `find()` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic* static method), 427
- `find()` (*rti.connexdds.PublicationBuiltinTopicData.Topic* static method), 457
- `find()` (*rti.connexdds.ServiceRequest.ContentFilteredTopic* static method), 525
- `find()` (*rti.connexdds.ServiceRequest.Topic* static method), 552
- `find()` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic* static method), 585
- `find()` (*rti.connexdds.SubscriptionBuiltinTopicData.Topic* static method), 615
- `find()` (*rti.connexdds.Topic* static method), 639
- `find()` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic* static method), 641
- `find()` (*rti.connexdds.TopicBuiltinTopicData.Topic* static method), 670
- `find()` (*rti.connexdds.TopicQuery* static method), 686
- `find_all_by_topic()` (*rti.connexdds.DataReader* static method), 87
- `find_all_by_topic()` (*rti.connexdds.DataWriter* static method), 116
- `find_all_by_topic()` (*rti.connexdds.DynamicData.DataReader* static method), 191
- `find_all_by_topic()` (*rti.connexdds.DynamicData.DataWriter* static method), 199
- `find_all_by_topic()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* static method), 377
- `find_all_by_topic()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter* static method), 385
- `find_all_by_topic()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader* static method), 433
- `find_all_by_topic()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter* static method), 442
- `find_all_by_topic()` (*rti.connexdds.ServiceRequest.DataReader* static method), 530
- `find_all_by_topic()` (*rti.connexdds.ServiceRequest.DataWriter* static method), 538
- `find_all_by_topic()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader* static method), 592
- `find_all_by_topic()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* static method), 600
- `find_all_by_topic()` (*rti.connexdds.TopicBuiltinTopicData.DataReader* static method), 647
- `find_all_by_topic()` (*rti.connexdds.TopicBuiltinTopicData.DataWriter* static method), 655
- `find_by_name()` (*rti.connexdds.DataReader* static method), 87
- `find_by_name()` (*rti.connexdds.DataWriter* static method), 116
- `find_by_name()` (*rti.connexdds.DynamicData.DataReader* static method), 191
- `find_by_name()` (*rti.connexdds.DynamicData.DataWriter* static method), 200
- `find_by_name()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* static method), 377
- `find_by_name()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter* static method), 386
- `find_by_name()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader* static method), 433
- `find_by_name()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter* static method), 442
- `find_by_name()` (*rti.connexdds.ServiceRequest.DataReader* static method), 530
- `find_by_name()` (*rti.connexdds.ServiceRequest.DataWriter* static method), 539
- `find_by_name()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader* static method),

592

`find_by_name()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter static method*), 600

`find_by_name()` (*rti.connexdds.TopicBuiltinTopicData.DataReader static method*), 647

`find_by_name()` (*rti.connexdds.TopicBuiltinTopicData.DataWriter static method*), 655

`find_by_topic()` (*rti.connexdds.DataReader static method*), 87

`find_by_topic()` (*rti.connexdds.DataWriter static method*), 116

`find_by_topic()` (*rti.connexdds.DynamicData.DataReader static method*), 191

`find_by_topic()` (*rti.connexdds.DynamicData.DataWriter static method*), 200

`find_by_topic()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader static method*), 377

`find_by_topic()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter static method*), 386

`find_by_topic()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader static method*), 433

`find_by_topic()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter static method*), 442

`find_by_topic()` (*rti.connexdds.ServiceRequest.DataReader static method*), 531

`find_by_topic()` (*rti.connexdds.ServiceRequest.DataWriter static method*), 539

`find_by_topic()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader static method*), 592

`find_by_topic()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter static method*), 601

`find_by_topic()` (*rti.connexdds.TopicBuiltinTopicData.DataReader static method*), 647

`find_by_topic()` (*rti.connexdds.TopicBuiltinTopicData.DataWriter static method*), 656

`find_contentfilter()` (*rti.connexdds.DomainParticipant method*), 160

`find_datareader()` (*rti.connexdds.DomainParticipant method*), 160

`find_datareader()` (*rti.connexdds.Subscriber method*), 579

`find_datareader_by_topic_name()` (*rti.connexdds.Subscriber method*), 579

`find_datareaders()` (*rti.connexdds.Subscriber method*), 579

`find_datawriter()` (*rti.connexdds.DomainParticipant method*), 160

`find_datawriter()` (*rti.connexdds.Publisher method*), 473

`find_datawriter_by_topic_name()` (*rti.connexdds.Publisher method*), 473

`find_datawriters()` (*rti.connexdds.Publisher method*), 473

`find_flow_controller()` (*rti.connexdds.DomainParticipant method*), 160

`find_member_by_id()` (*rti.connexdds.StructType method*), 578

`find_member_by_id()` (*rti.connexdds.UnionType method*), 741

`find_member_by_label()` (*rti.connexdds.UnionType method*), 741

`find_member_by_name()` (*rti.connexdds.ACTEnumMember method*), 32

`find_member_by_name()` (*rti.connexdds.ACTMember method*), 33

`find_member_by_name()` (*rti.connexdds.ACTUnionMember method*), 34

`find_member_by_ordinal()` (*rti.connexdds.EnumType method*), 266

`find_publisher()` (*rti.connexdds.DomainParticipant method*), 161

`find_publishers()` (*rti.connexdds.DomainParticipant method*), 161

`find_registered_content_filters()` (*rti.connexdds.DomainParticipant method*), 161

`find_subscriber()` (*rti.connexdds.DomainParticipant method*), 161

`find_subscribers()` (*rti.connexdds.DomainParticipant method*), 161

`find_topics()` (*rti.connexdds.DomainParticipant method*), 161

`first_available_sample_sequence_number` (*rti.connexdds.DataReaderProtocolStatus property*), 96

`first_available_sample_sequence_number` (*rti.connexdds.DataWriterProtocolStatus property*), 123

`first_available_sample_virtual_sequence_number` (*rti.connexdds.DataWriterProtocolStatus property*), 123

`first_unacknowledged_sample_sequence_number` (*rti.connexdds.DataWriterProtocolStatus property*), 123

`first_unacknowledged_sample_subscription_handle` (*rti.connexdds.DataWriterProtocolStatus property*), 123

`first_unacknowledged_sample_virtual_sequence_number` (*rti.connexdds.DataWriterProtocolStatus property*), 123

`first_unelapsed_keep_duration_sample_sequence_number` (*rti.connexdds.DataWriterProtocolStatus property*), 123

- FIXED_RATE_NAME (*rti.connexdds.FlowController* attribute), 280
- flag (*rti.connexdds.SampleInfo* property), 508
- flag (*rti.connexdds.WriteParams* property), 755
- flip () (*rti.connexdds.ActivityContextMask* method), 39
- flip () (*rti.connexdds.CompressionIdMask* method), 72
- flip () (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 152
- flip () (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 156
- flip () (*rti.connexdds.InstanceState* method), 312
- flip () (*rti.connexdds.RtpsReservedPortKindMask* method), 503
- flip () (*rti.connexdds.SampleFlag* method), 506
- flip () (*rti.connexdds.SampleLostState* method), 512
- flip () (*rti.connexdds.SampleRejectedState* method), 515
- flip () (*rti.connexdds.SampleState* method), 518
- flip () (*rti.connexdds.StatusMask* method), 568
- flip () (*rti.connexdds.StreamKind* method), 571
- flip () (*rti.connexdds.ThreadSettingsKindMask* method), 635
- flip () (*rti.connexdds.TransportBuiltinMask* method), 695
- flip () (*rti.connexdds.ViewState* method), 746
- Float32Seq (class in *rti.connexdds*), 274
- Float32Type (class in *rti.connexdds*), 276
- Float64Seq (class in *rti.connexdds*), 277
- Float64Type (class in *rti.connexdds*), 279
- Float128Seq (class in *rti.connexdds*), 272
- Float128Type (class in *rti.connexdds*), 274
- FLOAT_32_TYPE (*rti.connexdds.TypeKind* attribute), 720
- FLOAT_32_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 722
- FLOAT_64_TYPE (*rti.connexdds.TypeKind* attribute), 720
- FLOAT_64_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 722
- FLOAT_128_TYPE (*rti.connexdds.TypeKind* attribute), 720
- FLOAT_128_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 722
- FLOATING_POINT (*rti.connexdds.ThreadSettingsKindMask* attribute), 633
- FloatSeq (in module *rti.connexdds*), 279
- FloatType (in module *rti.connexdds*), 279
- flow_controller_allocation (*rti.connexdds.DomainParticipantResourceLimits* property), 172
- flow_controller_hash_buckets (*rti.connexdds.DomainParticipantResourceLimits* property), 172
- flow_controller_name (*rti.connexdds.PublishMode* property), 470
- FlowController (class in *rti.connexdds*), 279
- FlowControllerProperty (class in *rti.connexdds*), 280
- FlowControllerSchedulingPolicy (class in *rti.connexdds*), 281
- FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy (class in *rti.connexdds*), 281
- FlowControllerTokenBucketProperty (class in *rti.connexdds*), 284
- flush () (*rti.connexdds.DataWriter* method), 116
- flush () (*rti.connexdds.DynamicData.DataWriter* method), 200
- flush () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter* method), 386
- flush () (*rti.connexdds.PublicationBuiltinTopicData.DataWriter* method), 442
- flush () (*rti.connexdds.ServiceRequest.DataWriter* method), 539
- flush () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* method), 601
- flush () (*rti.connexdds.TopicBuiltinTopicData.DataWriter* method), 656
- force_type_validation (*rti.connexdds.TypeConsistencyEnforcement* property), 717
- from_cdr_buffer () (*rti.connexdds.DynamicData* method), 222
- from_cdr_buffer () (*rti.connexdds.DynamicData.TopicTypeSupport* static method), 218
- from_json () (*rti.connexdds.DynamicData* method), 222
- from_microseconds () (*rti.connexdds.Duration* static method), 184
- from_microseconds () (*rti.connexdds.Time* static method), 637
- from_milliseconds () (*rti.connexdds.Duration* static method), 184
- from_milliseconds () (*rti.connexdds.Time* static method), 637
- from_seconds () (*rti.connexdds.Duration* static method), 184
- from_seconds () (*rti.connexdds.Time* static method), 637
- from_string () (*rti.connexdds.DynamicData* method), 222
- FULL (*rti.connexdds.PersistentSynchronizationKind* attribute), 414
- full_reliable_writer_cache (*rti.connexdds.ReliableWriterCacheChangedStatus* property), 493
- FULLY_PROCESSED_INSTANCES (*rti.connexdds.DataReaderInstanceRemovalKind* attribute), 93
- FULLY_PROCESSED_INSTANCES (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind* attribute), 92
- function_name (*rti.connexdds.TransportMulticastMappingFunction* property), 706
- ## G
- generation (*rti.connexdds.Rank* property), 487
- generation_count (*rti.connexdds.SampleInfo* property), 508
- GenerationCount (class in *rti.connexdds*), 285

generic_auto_tuning (*rti.connextdds.BuiltinProfiles attribute*), 56
 generic_best_effort (*rti.connextdds.BuiltinProfiles attribute*), 56
 generic_common (*rti.connextdds.BuiltinProfiles attribute*), 56
 generic_connext_micro_compatibility (*rti.connextdds.BuiltinProfiles attribute*), 56
 generic_connext_micro_compatibility_2_4_3 (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_connext_micro_compatibility_2_4_9 (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_keep_last_reliable (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_keep_last_reliable_large_data (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_keep_last_reliable_large_data_fast_flow (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_keep_last_reliable_large_data_medium_flow (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_keep_last_reliable_large_data_slow_flow (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_keep_last_reliable_persistent (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_keep_last_reliable_transient (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_keep_last_reliable_transient_local (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_minimal_memory_footprint (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_monitoring_common (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_other_dds_vendor_compatibility (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_participant_large_data (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_participant_large_data_monitoring (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_security (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_strict_reliable (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_strict_reliable_high_throughput (*rti.connextdds.BuiltinProfiles attribute*), 57
 generic_strict_reliable_large_data (*rti.connextdds.BuiltinProfiles attribute*), 58
 generic_strict_reliable_large_data_fast_flow (*rti.connextdds.BuiltinProfiles attribute*), 58
 generic_strict_reliable_large_data_medium_flow (*rti.connextdds.BuiltinProfiles attribute*), 58
 generic_strict_reliable_large_data_slow_flow (*rti.connextdds.BuiltinProfiles attribute*), 58
 generic_strict_reliable_low_latency (*rti.connextdds.BuiltinProfiles attribute*), 58
 get () (*rti.connextdds.DataTag method*), 113
 get () (*rti.connextdds.Property method*), 424
 get_all () (*rti.connextdds.DataTag method*), 113
 get_all () (*rti.connextdds.Property method*), 424
 get_boolean () (*rti.connextdds.DynamicData method*), 223
 get_cdr_buffer () (*rti.connextdds.DynamicData method*), 223
 get_char () (*rti.connextdds.DynamicData method*), 223
 get_char_values () (*rti.connextdds.DynamicData method*), 223
 get_complex () (*rti.connextdds.DynamicData method*), 223
 get_complex_values () (*rti.connextdds.DynamicData method*), 223
 get_double () (*rti.connextdds.DynamicData method*), 224
 get_double_values () (*rti.connextdds.DynamicData method*), 224
 get_float () (*rti.connextdds.DynamicData method*), 224
 get_float32 () (*rti.connextdds.DynamicData method*), 224
 get_float32_values () (*rti.connextdds.DynamicData method*), 225
 get_float64 () (*rti.connextdds.DynamicData method*), 225
 get_float64_values () (*rti.connextdds.DynamicData method*), 225
 get_float128 () (*rti.connextdds.DynamicData method*), 224
 get_float_values () (*rti.connextdds.DynamicData method*), 225
 get_int () (*rti.connextdds.DynamicData method*), 225
 get_int8 () (*rti.connextdds.DynamicData method*), 227
 get_int16 () (*rti.connextdds.DynamicData method*), 226
 get_int16_values () (*rti.connextdds.DynamicData method*), 226
 get_int32 () (*rti.connextdds.DynamicData method*), 226
 get_int32_values () (*rti.connextdds.DynamicData method*), 226
 get_int64 () (*rti.connextdds.DynamicData method*), 226
 get_int64_values () (*rti.connextdds.DynamicData method*), 226
 get_int_values () (*rti.connextdds.DynamicData method*), 227
 get_long () (*rti.connextdds.DynamicData method*), 227
 get_long_values () (*rti.connextdds.DynamicData method*), 227
 get_longdouble () (*rti.connextdds.DynamicData method*), 227
 get_longlong () (*rti.connextdds.DynamicData method*), 228
 get_longlong_values () (*rti.connextdds.DynamicData method*), 228
 get_octet () (*rti.connextdds.DynamicData method*), 228
 get_octet_values () (*rti.connextdds.DynamicData method*), 228
 get_short () (*rti.connextdds.DynamicData method*), 228
 get_short_values () (*rti.connextdds.DynamicData method*), 228
 get_string () (*rti.connextdds.DynamicData method*), 229
 get_topic_datareader_qos () (*rti.connextdds.QosProvider method*), 482
 get_topic_datawriter_qos () (*rti.connextdds.QosProvider method*), 482

- get_topic_name_qos() (*rti.connexdds.QosProvider method*), 482
 get_uint() (*rti.connexdds.DynamicData method*), 229
 get_uint8() (*rti.connexdds.DynamicData method*), 230
 get_uint8_values() (*rti.connexdds.DynamicData method*), 230
 get_uint16() (*rti.connexdds.DynamicData method*), 229
 get_uint16_values() (*rti.connexdds.DynamicData method*), 229
 get_uint32() (*rti.connexdds.DynamicData method*), 229
 get_uint32_values() (*rti.connexdds.DynamicData method*), 230
 get_uint64() (*rti.connexdds.DynamicData method*), 230
 get_uint64_values() (*rti.connexdds.DynamicData method*), 230
 get_uint_values() (*rti.connexdds.DynamicData method*), 231
 get_ulong() (*rti.connexdds.DynamicData method*), 231
 get_ulong_values() (*rti.connexdds.DynamicData method*), 231
 get_ulonglong() (*rti.connexdds.DynamicData method*), 231
 get_ulonglong_values() (*rti.connexdds.DynamicData method*), 231
 get_ushort() (*rti.connexdds.DynamicData method*), 232
 get_ushort_values() (*rti.connexdds.DynamicData method*), 232
 get_value() (*rti.connexdds.DynamicData method*), 232
 get_values() (*rti.connexdds.DynamicData method*), 232
 get_wchar() (*rti.connexdds.DynamicData method*), 232
 get_wstring() (*rti.connexdds.DynamicData method*), 233
 GROUP (*rti.connexdds.PresentationAccessScopeKind attribute*), 416
 GROUP (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*), 416
 group_access_scope() (*rti.connexdds.Presentation static method*), 415
 group_coherent_set_sequence_number (*rti.connexdds.CoherentSetInfo property*), 69
 group_data (*rti.connexdds.PublicationBuiltinTopicData property*), 462
 group_data (*rti.connexdds.PublisherQos property*), 476
 group_data (*rti.connexdds.SubscriberQos property*), 582
 group_data (*rti.connexdds.SubscriptionBuiltinTopicData property*), 621
 group_guid (*rti.connexdds.CoherentSetInfo property*), 69
 GroupData (*class in rti.connexdds*), 285
 GuardCondition (*class in rti.connexdds*), 286
 Guid (*class in rti.connexdds*), 287
 guid (*rti.connexdds.TopicQuery property*), 686
 GUID_PREFIX (*rti.connexdds.ActivityContextMask attribute*), 38
- H**
- handle (*rti.connexdds.WriteParams property*), 755
 has_id (*rti.connexdds.UnionMember property*), 737
 has_parent (*rti.connexdds.StructType property*), 578
 heartbeat_period (*rti.connexdds.RtpsReliableWriterProtocol property*), 499
 heartbeat_suppression_duration (*rti.connexdds.RtpsReliableReaderProtocol property*), 497
 heartbeats_per_max_samples (*rti.connexdds.RtpsReliableWriterProtocol property*), 499
 high_watermark (*rti.connexdds.RtpsReliableWriterProtocol property*), 499
 high_watermark_reliable_writer_cache (*rti.connexdds.ReliableWriterCacheChangedStatus property*), 493
 HIGHEST_OFFERED (*rti.connexdds.PresentationAccessScopeKind attribute*), 416
 HIGHEST_OFFERED (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*), 416
 HIGHEST_PRIORITY_FIRST (*rti.connexdds.FlowControllerSchedulingPolicy attribute*), 283
 HIGHEST_PRIORITY_FIRST (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy attribute*), 282
 History (*class in rti.connexdds*), 288
 history (*rti.connexdds.DataReaderQos property*), 102
 history (*rti.connexdds.DataWriterQos property*), 131
 history (*rti.connexdds.TopicBuiltinTopicData property*), 675
 history (*rti.connexdds.TopicQos property*), 685
 history_depth (*rti.connexdds.DurabilityService property*), 182
 history_kind (*rti.connexdds.DurabilityService property*), 182
 HISTORY_SNAPSHOT (*rti.connexdds.TopicQuerySelectionKind attribute*), 688
 HISTORY_SNAPSHOT (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind attribute*), 688
 HistoryKind (*class in rti.connexdds*), 288
 HistoryKind.HistoryKind (*class in rti.connexdds*), 288
- I**
- IAnyDataReader (*class in rti.connexdds*), 290
 IAnyDataReaderSeq (*class in rti.connexdds*), 291
 IAnyDataWriter (*class in rti.connexdds*), 293
 IAnyDataWriterSeq (*class in rti.connexdds*), 294
 IAnyTopic (*class in rti.connexdds*), 296
 IAnyTopicSeq (*class in rti.connexdds*), 297
 ICondition (*class in rti.connexdds*), 299
 IConditionSeq (*class in rti.connexdds*), 299
 id (*rti.connexdds.Member property*), 346
 id (*rti.connexdds.UnionMember property*), 737
 IDataReader (*class in rti.connexdds*), 301
 identity (*rti.connexdds.WriteParams property*), 755
 IEntity (*class in rti.connexdds*), 302

IEntitySeq (class in *rti.connexdds*), 302
 ignore_dataareader () (*rti.connexdds.DomainParticipant method*), 161
 ignore_datareaders () (*rti.connexdds.DomainParticipant method*), 161
 ignore_datawriter () (*rti.connexdds.DomainParticipant method*), 161
 ignore_datawriters () (*rti.connexdds.DomainParticipant method*), 161
 ignore_default_domain_announcements (*rti.connexdds.DiscoveryConfig property*), 148
 ignore_enum_literal_names (*rti.connexdds.TypeConsistencyEnforcement property*), 717
 ignore_environment_profile (*rti.connexdds.QosProviderParams property*), 484
 ignore_member_names (*rti.connexdds.TypeConsistencyEnforcement property*), 717
 ignore_participant () (*rti.connexdds.DomainParticipant method*), 161
 ignore_participants () (*rti.connexdds.DomainParticipant method*), 161
 ignore_resource_profile (*rti.connexdds.QosProviderParams property*), 484
 ignore_sequence_bounds (*rti.connexdds.TypeConsistencyEnforcement property*), 717
 ignore_string_bounds (*rti.connexdds.TypeConsistencyEnforcement property*), 717
 ignore_topic () (*rti.connexdds.DomainParticipant method*), 161
 ignore_topics () (*rti.connexdds.DomainParticipant method*), 162
 ignore_user_profile (*rti.connexdds.QosProviderParams property*), 484
 ignored_entity_allocation (*rti.connexdds.DomainParticipantResourceLimits property*), 172
 ignored_entity_hash_buckets (*rti.connexdds.DomainParticipantResourceLimits property*), 172
 ignored_entity_replacement_kind (*rti.connexdds.DomainParticipantResourceLimits property*), 172
 IgnoredEntityReplacementKind (class in *rti.connexdds*), 305
 IgnoredEntityReplacementKind.IgnoredEntityReplacementKind (class in *rti.connexdds*), 305
 IllegalOperationError, 308
 ImmutablePolicyError, 308
 implicit_publisher (*rti.connexdds.DomainParticipant property*), 162
 implicit_subscriber (*rti.connexdds.DomainParticipant property*), 162
 inactivate_nonprogressing_readers (*rti.connexdds.RtpsReliableWriterProtocol property*), 499
 inactive_count (*rti.connexdds.ReliableReaderActivityChangedStatus property*), 492
 include_root_elements (*rti.connexdds.PrintFormatProperty property*), 423
 incomplete_coherent_set (*rti.connexdds.CoherentSetInfo property*), 69
 INCONSISTENT_TOPIC (*rti.connexdds.StatusMask attribute*), 566
 inconsistent_topic_status (*rti.connexdds.DynamicData.Topic property*), 215
 inconsistent_topic_status (*rti.connexdds.ParticipantBuiltinTopicData.Topic property*), 400
 inconsistent_topic_status (*rti.connexdds.PublicationBuiltinTopicData.Topic property*), 457
 inconsistent_topic_status (*rti.connexdds.ServiceRequest.Topic property*), 552
 inconsistent_topic_status (*rti.connexdds.SubscriptionBuiltinTopicData.Topic property*), 615
 inconsistent_topic_status (*rti.connexdds.Topic property*), 639
 inconsistent_topic_status (*rti.connexdds.TopicBuiltinTopicData.Topic property*), 670
 InconsistentPolicyError, 308
 InconsistentTopicStatus (class in *rti.connexdds*), 308
 incremental_count (*rti.connexdds.AllocationSettings property*), 41
 indent (*rti.connexdds.DynamicTypePrintFormatProperty property*), 256
 indent (*rti.connexdds.QosPrintFormat property*), 481
 index (*rti.connexdds.DynamicDataMemberInfo property*), 248
 infinite (*rti.connexdds.Duration attribute*), 184
 info (*rti.connexdds.DataReader.LoanedSample property*), 83
 info (*rti.connexdds.DynamicData property*), 233
 info (*rti.connexdds.DynamicData.LoanedSample property*), 210
 info (*rti.connexdds.DynamicData.Sample property*), 214
 info (*rti.connexdds.ParticipantBuiltinTopicData.LoanedSample property*), 395
 info (*rti.connexdds.ParticipantBuiltinTopicData.Sample property*), 399
 info (*rti.connexdds.PublicationBuiltinTopicData.LoanedSample property*), 452
 info (*rti.connexdds.PublicationBuiltinTopicData.Sample property*), 456
 info (*rti.connexdds.ServiceRequest.LoanedSample property*), 548
 info (*rti.connexdds.ServiceRequest.Sample property*), 551
 info (*rti.connexdds.SubscriptionBuiltinTopicData.LoanedSample property*), 610
 info (*rti.connexdds.SubscriptionBuiltinTopicData.Sample property*), 614

info (rti.connexdds.TopicBuiltinTopicData.LoanedSample property), 665
info (rti.connexdds.TopicBuiltinTopicData.Sample property), 669
INFO (rti.logging.distlog.LogLevel attribute), 771
info () (rti.logging.distlog.Logger static method), 771
INFORMATIONAL (rti.connexdds.SyslogVerbosity attribute), 629
informational () (rti.connexdds.Logger method), 344
init () (rti.logging.distlog.Logger static method), 771
initial_active_topic_queries (rti.connexdds.DataWriterResourceLimits property), 133
initial_batches (rti.connexdds.DataWriterResourceLimits property), 133
initial_buffer_size (rti.connexdds.DynamicDataProperty property), 249
initial_concurrent_blocking_threads (rti.connexdds.DataWriterResourceLimits property), 133
initial_count (rti.connexdds.Allocation.Settings property), 41
initial_count (rti.connexdds.Event property), 267
initial_fragmented_samples (rti.connexdds.BuiltinTopicReaderResourceLimits property), 59
initial_fragmented_samples (rti.connexdds.DataReaderResourceLimits property), 104
initial_infos (rti.connexdds.BuiltinTopicReaderResourceLimits property), 59
initial_infos (rti.connexdds.DataReaderResourceLimits property), 104
initial_instances (rti.connexdds.ResourceLimits property), 497
initial_objects_per_thread (rti.connexdds.SystemResourceLimits property), 630
initial_outstanding_reads (rti.connexdds.BuiltinTopicReaderResourceLimits property), 59
initial_outstanding_reads (rti.connexdds.DataReaderResourceLimits property), 104
initial_participant_announcements (rti.connexdds.DiscoveryConfig property), 148
initial_peers (rti.connexdds.Discovery property), 147
initial_records (rti.connexdds.Database property), 141
initial_remote_virtual_writers (rti.connexdds.DataReaderResourceLimits property), 105
initial_remote_virtual_writers_per_instance (rti.connexdds.DataReaderResourceLimits property), 105
initial_remote_writers (rti.connexdds.DataReaderResourceLimits property), 105
initial_remote_writers_per_instance (rti.connexdds.DataReaderResourceLimits property), 105
initial_samples (rti.connexdds.BuiltinTopicReaderResourceLimits property), 59
initial_samples (rti.connexdds.ResourceLimits property), 497
initial_topic_queries (rti.connexdds.DataReaderResourceLimits property), 105
initial_virtual_sequence_number (rti.connexdds.DataWriterProtocol property), 122
initial_virtual_writers (rti.connexdds.DataWriterResourceLimits property), 133
initial_weak_references (rti.connexdds.Database property), 141
initialize_sample () (rti.connexdds.DynamicData.TopicTypeSupport static method), 218
initialize_writer_loaned_sample (rti.connexdds.DataWriterResourceLimits property), 133
insert () (rti.connexdds.AnyDataReaderSeq method), 44
insert () (rti.connexdds.AnyDataWriterSeq method), 48
insert () (rti.connexdds.AnyTopicSeq method), 50
insert () (rti.connexdds.BoolSeq method), 55
insert () (rti.connexdds.ChannelSettingsSeq method), 65
insert () (rti.connexdds.CharSeq method), 67
insert () (rti.connexdds.ConditionSeq method), 75
insert () (rti.connexdds.ContentFilteredTopicSeq method), 79
insert () (rti.connexdds.CookieSeq method), 82
insert () (rti.connexdds.DataReaderSeq method), 109
insert () (rti.connexdds.DataWriterSeq method), 139
insert () (rti.connexdds.DomainParticipantSeq method), 177
insert () (rti.connexdds.DynamicData.ContentFilteredTopicSeq method), 188
insert () (rti.connexdds.DynamicData.DataReaderSeq method), 197
insert () (rti.connexdds.DynamicData.DataWriterSeq method), 208
insert () (rti.connexdds.DynamicDataSeq method), 251
insert () (rti.connexdds.DynamicData.TimestampedSeq method), 253
insert () (rti.connexdds.DynamicData.TopicSeq method), 218
insert () (rti.connexdds.EndpointGroupSeq method), 258
insert () (rti.connexdds.EntitySeq method), 262
insert () (rti.connexdds.EnumMemberSeq method), 265
insert () (rti.connexdds.Float32Seq method), 276
insert () (rti.connexdds.Float64Seq method), 279
insert () (rti.connexdds.Float128Seq method), 274
insert () (rti.connexdds.IAnyDataReaderSeq method), 293
insert () (rti.connexdds.IAnyDataWriterSeq method), 296
insert () (rti.connexdds.IAnyTopicSeq method), 299
insert () (rti.connexdds.IConditionSeq method), 301
insert () (rti.connexdds.IEntitySeq method), 304
insert () (rti.connexdds.InstanceHandleSeq method), 310
insert () (rti.connexdds.Int8Seq method), 325

- insert () (*rti.connexdds.Int16Seq method*), 316
- insert () (*rti.connexdds.Int32Seq method*), 319
- insert () (*rti.connexdds.Int64Seq method*), 322
- insert () (*rti.connexdds.LocatorFilterElementSeq method*), 335
- insert () (*rti.connexdds.LocatorSeq method*), 340
- insert () (*rti.connexdds.MemberSeq method*), 348
- insert () (*rti.connexdds.MonitoringMetricSelectionSeq method*), 355
- insert () (*rti.connexdds.MulticastMappingSeq method*), 360
- insert () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 373
- insert () (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 383
- insert () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 394
- insert () (*rti.connexdds.ParticipantBuiltinTopicDataSeq method*), 408
- insert () (*rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 411
- insert () (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq method*), 403
- insert () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 429
- insert () (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq method*), 439
- insert () (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq method*), 451
- insert () (*rti.connexdds.PublicationBuiltinTopicDataSeq method*), 466
- insert () (*rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq method*), 468
- insert () (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq method*), 460
- insert () (*rti.connexdds.PublisherSeq method*), 478
- insert () (*rti.connexdds.QosPolicyCountSeq method*), 480
- insert () (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq method*), 527
- insert () (*rti.connexdds.ServiceRequest.DataReaderSeq method*), 536
- insert () (*rti.connexdds.ServiceRequest.DataWriterSeq method*), 547
- insert () (*rti.connexdds.ServiceRequestSeq method*), 562
- insert () (*rti.connexdds.ServiceRequestTimestampedSeq method*), 564
- insert () (*rti.connexdds.ServiceRequest.TopicSeq method*), 555
- insert () (*rti.connexdds.StringPairSeq method*), 574
- insert () (*rti.connexdds.StringSeq method*), 576
- insert () (*rti.connexdds.SubscriberSeq method*), 584
- insert () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 588
- insert () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 597
- insert () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 609
- insert () (*rti.connexdds.SubscriptionBuiltinTopicDataSeq method*), 624
- insert () (*rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 627
- insert () (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq method*), 618
- insert () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 643
- insert () (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq method*), 652
- insert () (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method*), 664
- insert () (*rti.connexdds.TopicBuiltinTopicDataSeq method*), 678
- insert () (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq method*), 680
- insert () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq method*), 672
- insert () (*rti.connexdds.TopicSeq method*), 692
- insert () (*rti.connexdds.TransportInfoSeq method*), 701
- insert () (*rti.connexdds.TransportMulticastSeq method*), 708
- insert () (*rti.connexdds.TransportMulticastSettingsSeq method*), 711
- insert () (*rti.connexdds.TransportUnicastSettingsSeq method*), 715
- insert () (*rti.connexdds.Uint8Seq method*), 736
- insert () (*rti.connexdds.Uint16Seq method*), 728
- insert () (*rti.connexdds.Uint32Seq method*), 731
- insert () (*rti.connexdds.Uint64Seq method*), 733
- insert () (*rti.connexdds.UnionMemberSeq method*), 740
- insert () (*rti.connexdds.WcharSeq method*), 751
- insert () (*rti.connexdds.WstringSeq method*), 758
- INSTANCE (*rti.connexdds.DestinationOrderScopeKind attribute*), 145
- INSTANCE (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind attribute*), 145
- instance (*rti.connexdds.Logger attribute*), 344
- INSTANCE (*rti.connexdds.PresentationAccessScopeKind attribute*), 416
- INSTANCE (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind attribute*), 416

instance () (*rti.connexdds.DataReader.Selector method*), 84

instance () (*rti.connexdds.DynamicData.DataReader.Selector method*), 188

instance () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 374

instance () (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector method*), 430

instance () (*rti.connexdds.ServiceRequest.DataReader.Selector method*), 528

instance () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 589

instance () (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector method*), 644

instance_access_scope () (*rti.connexdds.Presentation static method*), 415

instance_handle (*rti.connexdds.IEntity property*), 302

instance_handle (*rti.connexdds.SampleInfo property*), 508

instance_hash_buckets (*rti.connexdds.ResourceLimits property*), 497

instance_id (*rti.connexdds.ServiceRequest property*), 557

instance_replacement (*rti.connexdds.DataReaderResourceLimits property*), 105

instance_replacement (*rti.connexdds.DataWriterResourceLimits property*), 133

instance_state (*rti.connexdds.DataState property*), 111

instance_state (*rti.connexdds.DataStateEx property*), 112

instance_state_consistency_kind (*rti.connexdds.Reliability property*), 490

InstanceHandle (*class in rti.connexdds*), 308

InstanceHandleSeq (*class in rti.connexdds*), 309

InstanceState (*class in rti.connexdds*), 311

InstanceStateConsistencyKind (*class in rti.connexdds*), 313

Int8Seq (*class in rti.connexdds*), 323

Int8Type (*class in rti.connexdds*), 325

Int16Seq (*class in rti.connexdds*), 314

Int16Type (*class in rti.connexdds*), 316

Int32Seq (*class in rti.connexdds*), 317

Int32Type (*class in rti.connexdds*), 319

Int32Vector (*class in rti.connexdds*), 319

Int64Seq (*class in rti.connexdds*), 320

Int64Type (*class in rti.connexdds*), 323

INT_16_TYPE (*rti.connexdds.TypeKind attribute*), 720

INT_16_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722

INT_32_TYPE (*rti.connexdds.TypeKind attribute*), 720

INT_32_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722

INT_64_TYPE (*rti.connexdds.TypeKind attribute*), 720

INT_64_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722

INTERMEDIATE_REPLY_SEQUENCE (*rti.connexdds.SampleFlag attribute*), 505

INTERMEDIATE_TOPIC_QUERY_SAMPLE (*rti.connexdds.SampleFlag attribute*), 505

interoperable (*rti.connexdds.RtpsWellKnownPorts attribute*), 504

INTRA (*rti.connexdds.LocatorKind attribute*), 335

INTRA (*rti.connexdds.LocatorKind.LocatorKind attribute*), 336

INTRA (*rti.connexdds.TransportClassId attribute*), 696

INTRA (*rti.connexdds.TransportClassId.TransportClassId attribute*), 697

invalid (*rti.connexdds.Locator attribute*), 331

INVALID (*rti.connexdds.LocatorKind attribute*), 335

INVALID (*rti.connexdds.LocatorKind.LocatorKind attribute*), 336

invalid (*rti.connexdds.Time attribute*), 637

INVALID (*rti.connexdds.TransportClassId attribute*), 696

INVALID (*rti.connexdds.TransportClassId.TransportClassId attribute*), 697

INVALID_ID (*rti.connexdds.Member attribute*), 346

INVALID_ID (*rti.connexdds.UnionMember attribute*), 737

INVALID_INDEX (*rti.connexdds.ACTEnumMember attribute*), 31

INVALID_INDEX (*rti.connexdds.ACTMember attribute*), 32

INVALID_INDEX (*rti.connexdds.ACTUnionMember attribute*), 33

INVALID_LOCAL_IDENTITY_ADVANCE_NOTICE (*rti.connexdds.StatusMask attribute*), 566

InvalidArgumentError, 325

InvalidDowncastError, 326

InvalidLocalIdentityAdvanceNoticeStatus (*class in rti.connexdds*), 326

IReadCondition (*class in rti.connexdds*), 304

is_aggregation_type () (*rti.connexdds.DynamicType method*), 255

is_cdr (*rti.connexdds.DynamicData property*), 233

is_collection_type () (*rti.connexdds.DynamicType method*), 255

is_constructed_type () (*rti.connexdds.DynamicType method*), 255

is_final_reply () (*rti.rpc.Requester class method*), 761

is_key (*rti.connexdds.Member property*), 346

is_keyed () (*rti.connexdds.DynamicType method*), 255

is_matched_publication_alive () (*rti.connexdds.DataReader method*), 87

is_matched_publication_alive () (*rti.connexdds.DynamicData.DataReader method*), 192

is_matched_publication_alive () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*), 377

is_matched_publication_alive () (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 433

is_matched_publication_alive () (*rti.connexdds.ServiceRequest.DataReader method*), 531

is_matched_publication_alive () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader*

- method), 592
- is_matched_publication_alive() (rti.connexdds.TopicBuiltinTopicData.DataReader method), 647
- is_matched_subscription_active() (rti.connexdds.DataWriter method), 116
- is_matched_subscription_active() (rti.connexdds.DynamicData.DataWriter method), 200
- is_matched_subscription_active() (rti.connexdds.ParticipantBuiltinTopicData.DataWriter method), 386
- is_matched_subscription_active() (rti.connexdds.PublicationBuiltinTopicData.DataWriter method), 442
- is_matched_subscription_active() (rti.connexdds.ServiceRequest.DataWriter method), 539
- is_matched_subscription_active() (rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method), 601
- is_matched_subscription_active() (rti.connexdds.TopicBuiltinTopicData.DataWriter method), 656
- is_member_key() (rti.connexdds.DynamicData method), 233
- is_nil (rti.connexdds.InstanceHandle property), 309
- is_optimized_storage (rti.connexdds.DynamicDataInfo property), 248
- is_pointer (rti.connexdds.AliasType property), 41
- is_primitive_type() (rti.connexdds.DynamicType method), 255
- is_related_reply() (rti.rpc.Requester class method), 761
- is_sample_app_acknowledged() (rti.connexdds.DataWriter method), 116
- is_sample_app_acknowledged() (rti.connexdds.DynamicData.DataWriter method), 200
- is_sample_app_acknowledged() (rti.connexdds.ParticipantBuiltinTopicData.DataWriter method), 386
- is_sample_app_acknowledged() (rti.connexdds.PublicationBuiltinTopicData.DataWriter method), 442
- is_sample_app_acknowledged() (rti.connexdds.ServiceRequest.DataWriter method), 539
- is_sample_app_acknowledged() (rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method), 601
- is_sample_app_acknowledged() (rti.connexdds.TopicBuiltinTopicData.DataWriter method), 656
- is_standalone (rti.connexdds.QosPrintFormat property), 481
- is_type_registered() (rti.connexdds.DomainParticipant method), 162
- items() (rti.connexdds.DynamicData method), 233
- items() (rti.connexdds.StringMap method), 572
- ITopicDescription (class in rti.connexdds), 305
- ## J
- journal_kind (rti.connexdds.PersistentStorageSettings property), 413
- JSON (rti.connexdds.PrintFormatKind attribute), 420
- JSON (rti.connexdds.PrintFormatKind.PrintFormatKind attribute), 421
- json (rti.connexdds.PrintFormatProperty attribute), 423
- ## K
- keep_all (rti.connexdds.History attribute), 288
- KEEP_ALL (rti.connexdds.HistoryKind attribute), 289
- KEEP_ALL (rti.connexdds.HistoryKind.HistoryKind attribute), 289
- KEEP_LAST (rti.connexdds.HistoryKind attribute), 289
- KEEP_LAST (rti.connexdds.HistoryKind.HistoryKind attribute), 289
- keep_last() (rti.connexdds.History static method), 288
- keep_minimum_state_for_instances (rti.connexdds.DataReaderResourceLimits property), 105
- key (rti.connexdds.ParticipantBuiltinTopicData property), 405
- key (rti.connexdds.PublicationBuiltinTopicData property), 462
- key (rti.connexdds.SubscriptionBuiltinTopicData property), 621
- key (rti.connexdds.TopicBuiltinTopicData property), 675
- key (rti.types.builtin.KeyedBytes attribute), 759
- key (rti.types.builtin.KeyedString attribute), 760
- key_only_filter (rti.connexdds.ExpressionProperty property), 269
- key_value() (rti.connexdds.DataReader method), 87
- key_value() (rti.connexdds.DataWriter method), 117
- key_value() (rti.connexdds.DynamicData.DataReader method), 192
- key_value() (rti.connexdds.DynamicData.DataWriter method), 200
- key_value() (rti.connexdds.ParticipantBuiltinTopicData.DataReader method), 377
- key_value() (rti.connexdds.ParticipantBuiltinTopicData.DataWriter method), 386
- key_value() (rti.connexdds.PublicationBuiltinTopicData.DataReader method), 434
- key_value() (rti.connexdds.PublicationBuiltinTopicData.DataWriter method), 442
- key_value() (rti.connexdds.ServiceRequest.DataReader method), 531
- key_value() (rti.connexdds.ServiceRequest.DataWriter method), 539
- key_value() (rti.connexdds.SubscriptionBuiltinTopicData.DataReader method), 592
- key_value() (rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method), 601

- key_value ()
(*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 647
- key_value ()
(*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 656
- KeyedBytes (class in *rti.types.builtin*), 759
- KeyedString (class in *rti.types.builtin*), 760
- keys () (*rti.connexdds.StringMap method*), 572
- kind (*rti.connexdds.DestinationOrder property*), 142
- kind (*rti.connexdds.Durability property*), 178
- kind (*rti.connexdds.DynamicDataMemberInfo property*), 248
- kind (*rti.connexdds.DynamicType property*), 255
- kind (*rti.connexdds.History property*), 288
- kind (*rti.connexdds.Liveliness property*), 327
- kind (*rti.connexdds.Locator property*), 331
- kind (*rti.connexdds.Ownership property*), 367
- kind (*rti.connexdds.PrintFormatProperty property*), 423
- kind (*rti.connexdds.PublishMode property*), 470
- kind (*rti.connexdds.Reliability property*), 490
- kind (*rti.connexdds.Service property*), 521
- kind (*rti.connexdds.TopicQuerySelection property*), 688
- kind (*rti.connexdds.TransportMulticast property*), 703
- kind (*rti.connexdds.TypeConsistencyEnforcement property*), 717
- L**
- label_count (*rti.connexdds.UnionMember property*), 737
- labels (*rti.connexdds.UnionMember property*), 738
- last_available_sample_sequence_number
(*rti.connexdds.DataReaderProtocolStatus property*), 96
- last_available_sample_sequence_number
(*rti.connexdds.DataWriterProtocolStatus property*), 123
- last_available_sample_virtual_sequence_number
(*rti.connexdds.DataWriterProtocolStatus property*), 123
- last_committed_sample_sequence_number
(*rti.connexdds.DataReaderProtocolStatus property*), 96
- last_corrupted_message_timestamp
(*rti.connexdds.DomainParticipantProtocolStatus property*), 166
- last_instance_handle
(*rti.connexdds.OfferedDeadlineMissedStatus property*), 366
- last_instance_handle
(*rti.connexdds.ReliableReaderActivityChangedStatus property*), 492
- last_instance_handle
(*rti.connexdds.RequestedDeadlineMissedStatus property*), 496
- last_instance_handle
(*rti.connexdds.SampleRejectedStatus property*), 516
- last_policy (*rti.connexdds.OfferedIncompatibleQosStatus property*), 366
- last_policy (*rti.connexdds.RequestedIncompatibleQosStatus property*), 496
- last_publication_handle
(*rti.connexdds.LivelinessChangedStatus property*), 328
- last_publication_handle
(*rti.connexdds.SubscriptionMatchedStatus property*), 627
- last_reason (*rti.connexdds.SampleLostStatus property*), 513
- last_reason (*rti.connexdds.SampleRejectedStatus property*), 516
- last_request_handle
(*rti.connexdds.ServiceRequestAcceptedStatus property*), 558
- LAST_SHARED_READER_QUEUE (*rti.connexdds.SampleFlag attribute*), 505
- last_subscription_handle
(*rti.connexdds.PublicationMatchedStatus property*), 469
- late_joiner_heartbeat_period
(*rti.connexdds.RtpsReliableWriterProtocol property*), 499
- latency_budget (*rti.connexdds.DataReaderQos property*), 103
- latency_budget (*rti.connexdds.DataWriterQos property*), 131
- latency_budget
(*rti.connexdds.PublicationBuiltinTopicData property*), 462
- latency_budget
(*rti.connexdds.SubscriptionBuiltinTopicData property*), 621
- latency_budget (*rti.connexdds.TopicBuiltinTopicData property*), 675
- latency_budget (*rti.connexdds.TopicQos property*), 685
- LatencyBudget (class in *rti.connexdds*), 326
- lease_duration (*rti.connexdds.Liveliness property*), 327
- length (*rti.connexdds.DataReader.LoanedSamples property*), 84
- length (*rti.connexdds.DynamicData.LoanedSamples property*), 211
- length (*rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples property*), 396
- length (*rti.connexdds.PublicationBuiltinTopicData.LoanedSamples property*), 452
- length (*rti.connexdds.ServiceRequest.LoanedSamples property*), 548
- length (*rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples property*), 611
- length (*rti.connexdds.TopicBuiltinTopicData.LoanedSamples property*), 665
- library_name (*rti.connexdds.BuiltinProfiles attribute*), 58
- Lifespan (class in *rti.connexdds*), 326
- lifespan (*rti.connexdds.DataWriterQos property*), 131
- lifespan (*rti.connexdds.PublicationBuiltinTopicData property*), 462

- lifespan (*rti.connextdds.TopicBuiltinTopicData* property), 675
- lifespan (*rti.connextdds.TopicQos* property), 685
- listener (*rti.connextdds.DataReader* property), 88
- listener (*rti.connextdds.DataWriter* property), 117
- listener (*rti.connextdds.DomainParticipant* property), 162
- listener (*rti.connextdds.DynamicData.DataReader* property), 192
- listener (*rti.connextdds.DynamicData.DataWriter* property), 200
- listener (*rti.connextdds.DynamicData.Topic* property), 215
- listener (*rti.connextdds.ParticipantBuiltinTopicData.DataReader* property), 377
- listener (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter* property), 386
- listener (*rti.connextdds.ParticipantBuiltinTopicData.Topic* property), 400
- listener (*rti.connextdds.PublicationBuiltinTopicData.DataReader* property), 434
- listener (*rti.connextdds.PublicationBuiltinTopicData.DataWriter* property), 443
- listener (*rti.connextdds.PublicationBuiltinTopicData.Topic* property), 457
- listener (*rti.connextdds.Publisher* property), 473
- listener (*rti.connextdds.ServiceRequest.DataReader* property), 531
- listener (*rti.connextdds.ServiceRequest.DataWriter* property), 539
- listener (*rti.connextdds.ServiceRequest.Topic* property), 552
- listener (*rti.connextdds.Subscriber* property), 579
- listener (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader* property), 592
- listener (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter* property), 601
- listener (*rti.connextdds.SubscriptionBuiltinTopicData.Topic* property), 615
- listener (*rti.connextdds.Topic* property), 639
- listener (*rti.connextdds.TopicBuiltinTopicData.DataReader* property), 647
- listener (*rti.connextdds.TopicBuiltinTopicData.DataWriter* property), 656
- listener (*rti.connextdds.TopicBuiltinTopicData.Topic* property), 670
- LIVE (*rti.connextdds.StreamKind* attribute), 569
- Liveliness (class in *rti.connextdds*), 327
- liveliness (*rti.connextdds.DataReaderQos* property), 103
- liveliness (*rti.connextdds.DataWriterQos* property), 131
- liveliness (*rti.connextdds.PublicationBuiltinTopicData* property), 462
- liveliness (*rti.connextdds.SubscriptionBuiltinTopicData* property), 621
- liveliness (*rti.connextdds.TopicBuiltinTopicData* property), 675
- liveliness (*rti.connextdds.TopicQos* property), 685
- LIVELINESS_BASED (*rti.connextdds.RemoteParticipantPurgeKind* attribute), 493
- LIVELINESS_BASED (*rti.connextdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* attribute), 494
- LIVELINESS_CHANGED (*rti.connextdds.StatusMask* attribute), 566
- liveliness_changed_status (*rti.connextdds.DataReader* property), 88
- liveliness_changed_status (*rti.connextdds.DynamicData.DataReader* property), 192
- liveliness_changed_status (*rti.connextdds.ParticipantBuiltinTopicData.DataReader* property), 377
- liveliness_changed_status (*rti.connextdds.PublicationBuiltinTopicData.DataReader* property), 434
- liveliness_changed_status (*rti.connextdds.ServiceRequest.DataReader* property), 531
- liveliness_changed_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataReader* property), 592
- liveliness_changed_status (*rti.connextdds.TopicBuiltinTopicData.DataReader* property), 647
- LIVELINESS_LOST (*rti.connextdds.StatusMask* attribute), 566
- liveliness_lost_status (*rti.connextdds.DataWriter* property), 117
- liveliness_lost_status (*rti.connextdds.DynamicData.DataWriter* property), 200
- liveliness_lost_status (*rti.connextdds.ParticipantBuiltinTopicData.DataWriter* property), 386
- liveliness_lost_status (*rti.connextdds.PublicationBuiltinTopicData.DataWriter* property), 443
- liveliness_lost_status (*rti.connextdds.ServiceRequest.DataWriter* property), 539
- liveliness_lost_status (*rti.connextdds.SubscriptionBuiltinTopicData.DataWriter* property), 601
- liveliness_lost_status (*rti.connextdds.TopicBuiltinTopicData.DataWriter* property), 656
- LivelinessChangedStatus (class in *rti.connextdds*), 328
- LivelinessKind (class in *rti.connextdds*), 328
- LivelinessKind.LivelinessKind (class in *rti.connextdds*), 328
- LivelinessLostStatus (class in *rti.connextdds*), 330
- load_profiles () (*rti.connextdds.QosProvider* method), 482
- loan_value () (*rti.connextdds.DynamicData* method), 233
- LoanedDynamicData (class in *rti.connextdds*), 331
- local_publisher_allocation (*rti.connextdds.DomainParticipantResourceLimits*

- `property`), 172
- `local_publisher_hash_buckets` (*rti.connexdds.DomainParticipantResourceLimits property*), 172
- `local_reader_allocation` (*rti.connexdds.DomainParticipantResourceLimits property*), 172
- `local_reader_hash_buckets` (*rti.connexdds.DomainParticipantResourceLimits property*), 172
- `local_subscriber_allocation` (*rti.connexdds.DomainParticipantResourceLimits property*), 172
- `local_subscriber_hash_buckets` (*rti.connexdds.DomainParticipantResourceLimits property*), 172
- `local_topic_allocation` (*rti.connexdds.DomainParticipantResourceLimits property*), 172
- `local_topic_hash_buckets` (*rti.connexdds.DomainParticipantResourceLimits property*), 172
- `local_writer_allocation` (*rti.connexdds.DomainParticipantResourceLimits property*), 173
- `local_writer_hash_buckets` (*rti.connexdds.DomainParticipantResourceLimits property*), 173
- `Locator` (*class in rti.connexdds*), 331
- `locator_filter` (*rti.connexdds.PublicationBuiltinTopicData property*), 462
- `locator_filters` (*rti.connexdds.LocatorFilter property*), 332
- `LOCATOR_REACHABILITY` (*rti.connexdds.ServiceRequestId attribute*), 558
- `LOCATOR_REACHABILITY` (*rti.connexdds.ServiceRequestId.ServiceRequestId attribute*), 558
- `locator_reachability_assert_period` (*rti.connexdds.DiscoveryConfig property*), 148
- `locator_reachability_change_detection_period` (*rti.connexdds.DiscoveryConfig property*), 148
- `locator_reachability_lease_duration` (*rti.connexdds.DiscoveryConfig property*), 148
- `LocatorFilter` (*class in rti.connexdds*), 332
- `LocatorFilterElement` (*class in rti.connexdds*), 332
- `LocatorFilterElementSeq` (*class in rti.connexdds*), 333
- `LocatorKind` (*class in rti.connexdds*), 335
- `LocatorKind.LocatorKind` (*class in rti.connexdds*), 335
- `locators` (*rti.connexdds.LocatorFilterElement property*), 333
- `LocatorSeq` (*class in rti.connexdds*), 338
- `LocatorVector` (*class in rti.connexdds*), 340
- `log()` (*rti.logging.distlog.Logger static method*), 771
- `log_infrastructure_messages` (*rti.logging.distlog.LoggerOptions property*), 772
- `log_level` (*rti.logging.distlog.MessageParams property*), 773
- `LogCategory` (*class in rti.connexdds*), 341
- `LogCategory.LogCategory` (*class in rti.connexdds*), 341
- `Logger` (*class in rti.connexdds*), 344
- `Logger` (*class in rti.logging.distlog*), 771
- `LoggerOptions` (*class in rti.logging.distlog*), 772
- `logging_settings` (*rti.connexdds.MonitoringDistributionSettings property*), 351
- `LogLevel` (*class in rti.logging.distlog*), 770
- `logs` (*rti.connexdds.MonitoringTelemetryData property*), 356
- `LongDouble` (*class in rti.connexdds*), 345
- `LongDoubleType` (*in module rti.connexdds*), 346
- `LongLongSeq` (*in module rti.connexdds*), 346
- `LongLongType` (*in module rti.connexdds*), 346
- `LongSeq` (*in module rti.connexdds*), 346
- `LongType` (*in module rti.connexdds*), 346
- `lookup_instance()` (*rti.connexdds.DataReader method*), 88
- `lookup_instance()` (*rti.connexdds.DataWriter method*), 117
- `lookup_instance()` (*rti.connexdds.DynamicData.DataReader method*), 192
- `lookup_instance()` (*rti.connexdds.DynamicData.DataWriter method*), 200
- `lookup_instance()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*), 377
- `lookup_instance()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*), 386
- `lookup_instance()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 434
- `lookup_instance()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*), 443
- `lookup_instance()` (*rti.connexdds.ServiceRequest.DataReader method*), 531
- `lookup_instance()` (*rti.connexdds.ServiceRequest.DataWriter method*), 539
- `lookup_instance()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*), 592
- `lookup_instance()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method*), 601
- `lookup_instance()` (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 647
- `lookup_instance()` (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 656
- `LOST_BY_AVAILABILITY_WAITING_TIME` (*rti.connexdds.SampleLostState attribute*), 510
- `LOST_BY_DECODE_FAILURE` (*rti.connexdds.SampleLostState attribute*), 510

LOST_BY_DESERIALIZATION_FAILURE
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_INCOMPLETE_COHERENT_SET
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_INSTANCES_LIMIT
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_LARGE_COHERENT_SET
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_OUT_OF_MEMORY (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_REMOTE_WRITERS_PER_INSTANCE_LIMIT
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_REMOTE_WRITERS_PER_SAMPLE_LIMIT
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_REMOTE_WRITERS_PER_VIRTUAL_QUEUE_LIMIT
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_SAMPLES_LIMIT (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_SAMPLES_PER_INSTANCE_LIMIT
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_SAMPLES_PER_REMOTE_WRITER_LIMIT
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_UNKNOWN_INSTANCE
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_VIRTUAL_WRITERS_LIMIT
 (*rti.connexdds.SampleLostState attribute*), 510
 LOST_BY_WRITER (*rti.connexdds.SampleLostState attribute*), 510
 low_watermark (*rti.connexdds.RtpsReliableWriterProtocol property*), 499
 low_watermark_reliable_writer_cache
 (*rti.connexdds.ReliableWriterCacheChangedStatus property*), 493
 LZ4 (*rti.connexdds.CompressionIdMask attribute*), 70

M

major_version (*rti.connexdds.ProductVersion property*), 423
 major_version (*rti.connexdds.ProtocolVersion property*), 425
 MANUAL_BY_PARTICIPANT (*rti.connexdds.LivelinessKind attribute*), 329
 MANUAL_BY_PARTICIPANT
 (*rti.connexdds.LivelinessKind.LivelinessKind attribute*), 329
 manual_by_participant() (*rti.connexdds.Liveliness static method*), 328
 MANUAL_BY_TOPIC (*rti.connexdds.LivelinessKind attribute*), 329
 MANUAL_BY_TOPIC
 (*rti.connexdds.LivelinessKind.LivelinessKind attribute*), 329
 manual_by_topic() (*rti.connexdds.Liveliness static method*), 328
 manually_enable (*rti.connexdds.EntityFactory attribute*), 260
 MAP_TYPE (*rti.connexdds.TypeKind attribute*), 720
 MAP_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722
 mapping_function (*rti.connexdds.MulticastMapping property*), 358
 mask (*rti.connexdds.ThreadSettings property*), 631
 mask (*rti.connexdds.TransportBuiltin property*), 693
 matched_client_count (*rti.rpc.Service property*), 768
 matched_publication_data()
 (*rti.connexdds.DataReader method*), 88
 matched_publication_data()
 (*rti.connexdds.DynamicData.DataReader method*), 192
 matched_publication_data() (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*), 378
 matched_publication_data() (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 434
 matched_publication_data()
 (*rti.connexdds.ServiceRequest.DataReader method*), 531
 matched_publication_data() (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*), 592
 matched_publication_data()
 (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 647
 matched_publication_datareader_protocol_status() (*rti.connexdds.DataReader method*), 88
 matched_publication_datareader_protocol_status()
 (*rti.connexdds.DynamicData.DataReader method*), 192
 matched_publication_datareader_protocol_status()
 (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*), 378
 matched_publication_datareader_protocol_status()
 (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 434
 matched_publication_datareader_protocol_status()
 (*rti.connexdds.ServiceRequest.DataReader method*), 531
 matched_publication_datareader_protocol_status()
 (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*), 592
 matched_publication_datareader_protocol_status()
 (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 648
 matched_publication_participant_data()
 (*rti.connexdds.DataReader method*), 88
 matched_publication_participant_data()

(*rti.connexdds.DynamicData.DataReader method*),
 192
matched_publication_participant_data()
 (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*),
 378
matched_publication_participant_data()
 (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*),
 434
matched_publication_participant_data()
 (*rti.connexdds.ServiceRequest.DataReader method*),
 531
matched_publication_participant_data()
 (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*),
 593
matched_publication_participant_data()
 (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 648
matched_publications (*rti.connexdds.DataReader property*), 88
matched_publications
 (*rti.connexdds.DynamicData.DataReader property*),
 192
matched_publications (*rti.connexdds.ParticipantBuiltinTopicData.DataReader property*),
 378
matched_publications (*rti.connexdds.PublicationBuiltinTopicData.DataReader property*),
 434
matched_publications
 (*rti.connexdds.ServiceRequest.DataReader property*),
 532
matched_publications (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader property*),
 593
matched_publications
 (*rti.connexdds.TopicBuiltinTopicData.DataReader property*), 648
matched_replier_count (*rti.rpc.Requester property*), 762
matched_requester_count (*rti.rpc.Replier property*), 765
matched_requester_count (*rti.rpc.SimpleReplier property*), 768
matched_service_count (*rti.rpc.ClientBase property*),
 769
matched_subscription_data()
 (*rti.connexdds.DataWriter method*), 117
matched_subscription_data()
 (*rti.connexdds.DynamicData.DataWriter method*),
 200
matched_subscription_data() (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*),
 386
matched_subscription_data() (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*),
 443
matched_subscription_data()
 (*rti.connexdds.ServiceRequest.DataWriter method*),
 540
matched_subscription_data()
 (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method*),
 602
matched_subscription_data()
 (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 656
matched_subscription_datawriter_protocol_status() (*rti.connexdds.DataWriter method*), 117
matched_subscription_datawriter_protocol_status()
 (*rti.connexdds.DynamicData.DataWriter method*),
 200
matched_subscription_datawriter_protocol_status()
 (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*),
 386
matched_subscription_datawriter_protocol_status()
 (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*),
 443
matched_subscription_datawriter_protocol_status()
 (*rti.connexdds.ServiceRequest.DataWriter method*),
 539
matched_subscription_datawriter_protocol_status()
 (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method*),
 601
matched_subscription_datawriter_protocol_status()
 (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 656
matched_subscription_participant_data()
 (*rti.connexdds.DataWriter method*), 117
matched_subscription_participant_data()
 (*rti.connexdds.DynamicData.DataWriter method*),
 201
matched_subscription_participant_data()
 (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*),
 387
matched_subscription_participant_data()
 (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*),
 443
matched_subscription_participant_data()
 (*rti.connexdds.ServiceRequest.DataWriter method*),
 540
matched_subscription_participant_data()
 (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method*),
 602
matched_subscription_participant_data()
 (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 656

- method*), 656
- `matched_subscriptions` (*rti.connexdds.DataWriter property*), 117
- `matched_subscriptions` (*rti.connexdds.DynamicData.DataWriter property*), 201
- `matched_subscriptions` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter property*), 387
- `matched_subscriptions` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter property*), 443
- `matched_subscriptions` (*rti.connexdds.ServiceRequest.DataWriter property*), 540
- `matched_subscriptions` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter property*), 602
- `matched_subscriptions` (*rti.connexdds.TopicBuiltinTopicData.DataWriter property*), 657
- `matched_subscriptions_locators` (*rti.connexdds.DataWriter property*), 117
- `matched_subscriptions_locators` (*rti.connexdds.DynamicData.DataWriter property*), 201
- `matched_subscriptions_locators` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter property*), 387
- `matched_subscriptions_locators` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter property*), 443
- `matched_subscriptions_locators` (*rti.connexdds.ServiceRequest.DataWriter property*), 540
- `matched_subscriptions_locators` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter property*), 602
- `matched_subscriptions_locators` (*rti.connexdds.TopicBuiltinTopicData.DataWriter property*), 657
- `matching_reader_writer_pair_allocation` (*rti.connexdds.DomainParticipantResourceLimits property*), 173
- `matching_reader_writer_pair_hash_buckets` (*rti.connexdds.DomainParticipantResourceLimits property*), 173
- `matching_writer_reader_pair_allocation` (*rti.connexdds.DomainParticipantResourceLimits property*), 173
- `matching_writer_reader_pair_hash_buckets` (*rti.connexdds.DomainParticipantResourceLimits property*), 173
- `max_active_topic_queries` (*rti.connexdds.DataWriterResourceLimits property*), 133
- `max_app_ack_remote_readers` (*rti.connexdds.DataWriterResourceLimits property*), 133
- `max_app_ack_response_length` (*rti.connexdds.DataReaderResourceLimits property*), 105
- `max_batches` (*rti.connexdds.DataWriterResourceLimits property*), 133
- `max_blocking_time` (*rti.connexdds.Reliability property*), 490
- `max_buffer_size` (*rti.connexdds.DynamicDataProperty property*), 249
- `max_buffer_size_increment` (*rti.connexdds.DynamicDataProperty property*), 249
- `max_bytes_per_nack_response` (*rti.connexdds.RtpsReliableWriterProtocol property*), 499
- `max_concurrent_blocking_threads` (*rti.connexdds.DataWriterResourceLimits property*), 133
- `max_count` (*rti.connexdds.AllocationSettings property*), 42
- `max_count` (*rti.connexdds.Event property*), 267
- `max_data_availability_waiting_time` (*rti.connexdds.Availability property*), 53
- `max_data_bytes` (*rti.connexdds.Batch property*), 53
- `max_endpoint_availability_waiting_time` (*rti.connexdds.Availability property*), 53
- `max_endpoint_group_cumulative_characters` (*rti.connexdds.DomainParticipantResourceLimits property*), 173
- `max_endpoint_groups` (*rti.connexdds.DomainParticipantResourceLimits property*), 173
- `max_flush_delay` (*rti.connexdds.Batch property*), 53
- `max_fragmented_samples` (*rti.connexdds.BuiltinTopicReaderResourceLimits property*), 59
- `max_fragmented_samples` (*rti.connexdds.DataReaderResourceLimits property*), 105
- `max_fragmented_samples_per_remote_writer` (*rti.connexdds.BuiltinTopicReaderResourceLimits property*), 59
- `max_fragmented_samples_per_remote_writer` (*rti.connexdds.DataReaderResourceLimits property*), 105
- `max_fragments_per_sample` (*rti.connexdds.BuiltinTopicReaderResourceLimits property*), 59
- `max_fragments_per_sample` (*rti.connexdds.DataReaderResourceLimits property*), 105
- `max_gather_destinations` (*rti.connexdds.DomainParticipantResourceLimits property*), 173
- `max_heartbeat_response_delay` (*rti.connexdds.RtpsReliableReaderProtocol property*), 497
- `max_heartbeat_retries` (*rti.connexdds.RtpsReliableWriterProtocol property*), 499
- `max_historical_logs`

(rti.connexdds.MonitoringLoggingDistributionSettings property), 352
max_infos (*rti.connexdds.BuiltinTopicReaderResourceLimits property*), 59
max_infos (*rti.connexdds.DataReaderResourceLimits property*), 105
max_initial_participant_announcement_period (*rti.connexdds.DiscoveryConfig property*), 149
max_instances (*rti.connexdds.DurabilityService property*), 182
max_instances (*rti.connexdds.ResourceLimits property*), 497
max_liveliness_loss_detection_period (*rti.connexdds.DiscoveryConfig property*), 149
max_nack_response_delay (*rti.connexdds.RtpsReliableWriterProtocol property*), 499
max_objects_per_thread (*rti.connexdds.SystemResourceLimits property*), 630
max_outstanding_reads (*rti.connexdds.BuiltinTopicReaderResourceLimits property*), 60
max_outstanding_reads (*rti.connexdds.DataReaderResourceLimits property*), 105
max_partition_cumulative_characters (*rti.connexdds.DomainParticipantResourceLimits property*), 173
max_partitions (*rti.connexdds.DomainParticipantResourceLimits property*), 173
max_query_condition_filters (*rti.connexdds.DataReaderResourceLimits property*), 105
max_remote_reader_filters (*rti.connexdds.DataWriterResourceLimits property*), 133
max_remote_readers (*rti.connexdds.DataWriterResourceLimits property*), 133
max_remote_virtual_writers (*rti.connexdds.DataReaderResourceLimits property*), 105
max_remote_virtual_writers_per_instance (*rti.connexdds.DataReaderResourceLimits property*), 106
max_remote_writers (*rti.connexdds.DataReaderResourceLimits property*), 106
max_remote_writers_per_instance (*rti.connexdds.DataReaderResourceLimits property*), 106
max_remote_writers_per_sample (*rti.connexdds.DataReaderResourceLimits property*), 106
max_samples (*rti.connexdds.Batch property*), 53
max_samples (*rti.connexdds.BuiltinTopicReaderResourceLimits property*), 60
max_samples (*rti.connexdds.DurabilityService property*), 182
max_samples (*rti.connexdds.ResourceLimits property*), 497
max_samples () (*rti.connexdds.DataReader.Selector method*), 84
max_samples () (*rti.connexdds.DynamicData.DataReader.Selector method*), 188
max_samples () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 374
max_samples () (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector method*), 430
max_samples () (*rti.connexdds.ServiceRequest.DataReader.Selector method*), 528
max_samples () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 589
max_samples () (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector method*), 644
max_samples_per_instance (*rti.connexdds.DurabilityService property*), 182
max_samples_per_instance (*rti.connexdds.ResourceLimits property*), 497
max_samples_per_read (*rti.connexdds.BuiltinTopicReaderResourceLimits property*), 60
max_samples_per_read (*rti.connexdds.DataReaderResourceLimits property*), 106
max_samples_per_remote_writer (*rti.connexdds.DataReaderResourceLimits property*), 106
max_send_window_size (*rti.connexdds.RtpsReliableWriterProtocol property*), 500
max_serialized_size (*rti.connexdds.DynamicDataTypeSerializationProperty property*), 254
max_skiplist_level (*rti.connexdds.Database property*), 141
max_tokens (*rti.connexdds.FlowControllerTokenBucketProperty property*), 285
max_topic_queries (*rti.connexdds.DataReaderResourceLimits property*), 106
max_total_instances (*rti.connexdds.DataReaderResourceLimits property*), 106
max_virtual_writers (*rti.connexdds.DataWriterResourceLimits property*), 133
max_weak_references (*rti.connexdds.Database property*), 141
MAXIMAL (*rti.connexdds.PrintFormat attribute*), 418

- MAXIMAL (*rti.connexdds.PrintFormat.PrintFormat attribute*), 419
- maximum (*rti.connexdds.SequenceNumber attribute*), 520
- maximum (*rti.connexdds.Time attribute*), 637
- Member (*class in rti.connexdds*), 346
- member () (*rti.connexdds.ACTEnumMember method*), 32
- member () (*rti.connexdds.ACTMember method*), 33
- member () (*rti.connexdds.ACTUnionMember method*), 34
- member_count (*rti.connexdds.ACTEnumMember property*), 32
- member_count (*rti.connexdds.ACTMember property*), 33
- member_count (*rti.connexdds.ACTUnionMember property*), 34
- member_count (*rti.connexdds.DynamicData property*), 233
- member_count (*rti.connexdds.DynamicDataInfo property*), 248
- member_exists () (*rti.connexdds.DynamicData method*), 233
- member_exists_in_type () (*rti.connexdds.DynamicData method*), 234
- member_index () (*rti.connexdds.DynamicData method*), 234
- member_info () (*rti.connexdds.DynamicData method*), 234
- members () (*rti.connexdds.ACTEnumMember method*), 32
- members () (*rti.connexdds.ACTMember method*), 33
- members () (*rti.connexdds.ACTUnionMember method*), 34
- MemberSeq (*class in rti.connexdds*), 347
- MEMORY (*rti.connexdds.PersistentJournalKind attribute*), 412
- message (*rti.logging.distlog.MessageParams property*), 773
- message_size_max (*rti.connexdds.TransportInfo property*), 699
- MessageParams (*class in rti.logging.distlog*), 773
- metatraffic_transport_priority (*rti.connexdds.Discovery property*), 147
- metrics (*rti.connexdds.MonitoringTelemetryData property*), 357
- middleware_forwarding_level (*rti.connexdds.MonitoringLoggingForwardingSettings property*), 352
- min_heartbeat_response_delay (*rti.connexdds.RtpsReliableReaderProtocol property*), 498
- min_initial_participant_announcement_period (*rti.connexdds.DiscoveryConfig property*), 149
- min_nack_response_delay (*rti.connexdds.RtpsReliableWriterProtocol property*), 500
- min_send_window_size (*rti.connexdds.RtpsReliableWriterProtocol property*), 500
- min_serialized_size (*rti.connexdds.DynamicDataType-SerializationProperty property*), 254
- min_serialized_size (*rti.connexdds.DynamicTypePrintFormatProperty property*), 256
- MINIMAL (*rti.connexdds.PrintFormat attribute*), 418
- MINIMAL (*rti.connexdds.PrintFormat.PrintFormat attribute*), 419
- minimum_separation (*rti.connexdds.TimeBasedFilter property*), 638
- minor_version (*rti.connexdds.ProductVersion property*), 423
- minor_version (*rti.connexdds.ProtocolVersion property*), 426
- module
- rti.asyncio, 770
 - rti.connexdds, 31
 - rti.logging, 773
 - rti.logging.distlog, 770
 - rti.logging.handler, 773
 - rti.types.builtin, 759
- Monitoring (*class in rti.connexdds*), 349
- monitoring (*rti.connexdds.DomainParticipantFactoryQos property*), 165
- MONITORING_LIBRARY_COMMAND (*rti.connexdds.ServiceRequestId attribute*), 558
- MONITORING_LIBRARY_COMMAND (*rti.connexdds.ServiceRequestId.ServiceRequestId attribute*), 559
- MONITORING_LIBRARY_REPLY (*rti.connexdds.ServiceRequestId attribute*), 558
- MONITORING_LIBRARY_REPLY (*rti.connexdds.ServiceRequestId.ServiceRequestId attribute*), 559
- MonitoringDedicatedParticipantSettings (*class in rti.connexdds*), 349
- MonitoringDistributionSettings (*class in rti.connexdds*), 350
- MonitoringEventDistributionSettings (*class in rti.connexdds*), 351
- MonitoringLoggingDistributionSettings (*class in rti.connexdds*), 351
- MonitoringLoggingForwardingSettings (*class in rti.connexdds*), 352
- MonitoringMetricSelection (*class in rti.connexdds*), 353
- MonitoringMetricSelectionSeq (*class in rti.connexdds*), 353
- MonitoringPeriodicDistributionSettings (*class in rti.connexdds*), 356
- MonitoringTelemetryData (*class in rti.connexdds*), 356
- multi_channel (*rti.connexdds.DataWriterQos property*), 131
- multicast_locators (*rti.connexdds.SubscriptionBuiltinTopicData property*), 621
- multicast_receive_addresses (*rti.connexdds.Discovery property*), 147
- multicast_resend_threshold (*rti.connexdds.RtpsReliableWriterProtocol property*), 500
- multicast_settings (*rti.connexdds.ChannelSettings property*), 63
- MulticastMapping (*class in rti.connexdds*), 357
- MulticastMappingSeq (*class in rti.connexdds*), 358
- MultiChannel (*class in rti.connexdds*), 357
- MUTABLE (*rti.connexdds.ExtensibilityKind attribute*), 270

MUTABLE (*rti.connexdds.ExtensibilityKind.ExtensibilityKind* attribute), 269

N

- nack_period* (*rti.connexdds.RtpsReliableReaderProtocol* property), 498
- nack_suppression_duration* (*rti.connexdds.RtpsReliableWriterProtocol* property), 500
- name* (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind* property), 36
- name* (*rti.connexdds.CdrPaddingKind.CdrPaddingKind* property), 62
- name* (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind* property), 93
- name* (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind* property), 136
- name* (*rti.connexdds.DestinationOrderKind.DestinationOrderKind* property), 143
- name* (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind* property), 145
- name* (*rti.connexdds.DurabilityKind.DurabilityKind* property), 180
- name* (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind* property), 246
- name* (*rti.connexdds.DynamicData.ITopicDescription* property), 209
- name* (*rti.connexdds.DynamicDataMemberInfo* property), 248
- name* (*rti.connexdds.DynamicType* property), 255
- name* (*rti.connexdds.EntityName* property), 261
- name* (*rti.connexdds.EnumMember* property), 263
- name* (*rti.connexdds.ExtensibilityKind.ExtensibilityKind* property), 270
- name* (*rti.connexdds.Filter* property), 271
- name* (*rti.connexdds.FlowController* property), 280
- name* (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy* property), 283
- name* (*rti.connexdds.HistoryKind.HistoryKind* property), 289
- name* (*rti.connexdds.IAnyTopic* property), 297
- name* (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind* property), 306
- name* (*rti.connexdds.InstanceStateConsistencyKind* property), 314
- name* (*rti.connexdds.ITopicDescription* property), 305
- name* (*rti.connexdds.LivelinessKind.LivelinessKind* property), 329
- name* (*rti.connexdds.LocatorKind.LocatorKind* property), 337
- name* (*rti.connexdds.LogCategory.LogCategory* property), 342
- name* (*rti.connexdds.Member* property), 346
- name* (*rti.connexdds.OwnershipKind.OwnershipKind* property), 368
- name* (*rti.connexdds.ParticipantBuiltinTopicData.ITopicDescription* property), 395
- name* (*rti.connexdds.Partition* property), 411
- name* (*rti.connexdds.PersistentJournalKind* property), 413
- name* (*rti.connexdds.PersistentSynchronizationKind* property), 415
- name* (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind* property), 417
- name* (*rti.connexdds.PrintFormatKind.PrintFormatKind* property), 421
- name* (*rti.connexdds.PrintFormat.PrintFormat* property), 419
- name* (*rti.connexdds.PublicationBuiltinTopicData.ITopicDescription* property), 451
- name* (*rti.connexdds.PublishModeKind.PublishModeKind* property), 471
- name* (*rti.connexdds.Query* property), 485
- name* (*rti.connexdds.ReliabilityKind.ReliabilityKind* property), 491
- name* (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* property), 494
- name* (*rti.connexdds.ServiceKind.ServiceKind* property), 522
- name* (*rti.connexdds.ServiceRequestId.ServiceRequestId* property), 559
- name* (*rti.connexdds.ServiceRequest.ITopicDescription* property), 547
- name* (*rti.connexdds.SubscriptionBuiltinTopicData.ITopicDescription* property), 610
- name* (*rti.connexdds.SyslogVerbosity* property), 629
- name* (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* property), 632
- name* (*rti.connexdds.TopicBuiltinTopicData* property), 675
- name* (*rti.connexdds.TopicBuiltinTopicData.ITopicDescription* property), 664
- name* (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind* property), 689
- name* (*rti.connexdds.TransportClassId.TransportClassId* property), 698
- name* (*rti.connexdds.TransportMulticastKind.TransportMulticastKind* property), 704
- name* (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind* property), 719
- name* (*rti.connexdds.TypeKind.TypeKind* property), 724
- name* (*rti.connexdds.UnionMember* property), 738
- name* (*rti.connexdds.Verbosity.Verbosity* property), 743
- name* (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* property), 754
- name* (*rti.logging.distlog.LogLevel* property), 771

- nanosec (*rti.connexdds.Duration* property), 184
- nanosec (*rti.connexdds.Time* property), 637
- native_build_id (*rti.connexdds.ProductVersion* property), 423
- new_data (*rti.connexdds.DataState* attribute), 111
- new_data (*rti.connexdds.DataStateEx* attribute), 112
- new_instance (*rti.connexdds.DataState* attribute), 111
- new_instance (*rti.connexdds.DataStateEx* attribute), 112
- new_remote_participant_announcements (*rti.connexdds.DiscoveryConfig* property), 149
- NEW_VIEW (*rti.connexdds.ViewState* attribute), 744
- next_instance () (*rti.connexdds.DataReader.Selector* method), 84
- next_instance () (*rti.connexdds.DynamicData.DataReader.Selector* method), 189
- next_instance () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector* method), 374
- next_instance () (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector* method), 430
- next_instance () (*rti.connexdds.ServiceRequest.DataReader.Selector* method), 528
- next_instance () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector* method), 589
- next_instance () (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector* method), 644
- nil () (*rti.connexdds.InstanceHandle* static method), 309
- no_auto_purge (*rti.connexdds.ReaderDataLifecycle* attribute), 489
- NO_INSTANCE (*rti.connexdds.DataReaderInstanceRemovalKind* attribute), 93
- NO_INSTANCE (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind* attribute), 92
- NO_PURGE (*rti.connexdds.RemoteParticipantPurgeKind* attribute), 493
- NO_PURGE (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* attribute), 494
- NO_REPLACEMENT (*rti.connexdds.IgnoredEntityReplacementKind* attribute), 307
- NO_REPLACEMENT (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind* attribute), 306
- NO_ROTATION (*rti.connexdds.ThreadSettingsCpuRotationKind* attribute), 631
- NO_ROTATION (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* attribute), 631
- NO_SERVICE (*rti.connexdds.ServiceKind* attribute), 521
- NO_SERVICE (*rti.connexdds.ServiceKind.ServiceKind* attribute), 522
- NO_TYPE (*rti.connexdds.TypeKind* attribute), 720
- NO_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 722
- no_writers (*rti.connexdds.GenerationCount* property), 285
- no_writers_instance_count (*rti.connexdds.DataReaderCacheStatus* property), 91
- no_writers_instance_count_peak (*rti.connexdds.DataReaderCacheStatus* property), 91
- no_writers_instance_removal (*rti.connexdds.DataReaderResourceLimitsInstanceReplacementSettings* property), 107
- NONE (*rti.connexdds.ActivityContextMask* attribute), 38
- NONE (*rti.connexdds.CompressionIdMask* attribute), 70
- NONE (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* attribute), 151
- NONE (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* attribute), 154
- NONE (*rti.connexdds.InstanceStateConsistencyKind* attribute), 314
- NONE (*rti.connexdds.RtpsReservedPortKindMask* attribute), 500
- NONE (*rti.connexdds.StatusMask* attribute), 566
- none (*rti.connexdds.TransportBuiltin* attribute), 693
- NONE (*rti.connexdds.TransportBuiltinMask* attribute), 693
- NoOpAnyDataReaderListener (*class in rti.connexdds*), 360
- NoOpAnyDataWriterListener (*class in rti.connexdds*), 361
- NoOpAnyTopicListener (*class in rti.connexdds*), 362
- NoOpDataReaderListener (*class in rti.connexdds*), 362
- NoOpDataWriterListener (*class in rti.connexdds*), 363
- NoOpDomainParticipantListener (*class in rti.connexdds*), 364
- NoOpPublisherListener (*class in rti.connexdds*), 365
- NoOpSubscriberListener (*class in rti.connexdds*), 365
- NoOpTopicListener (*class in rti.connexdds*), 365
- NORMAL (*rti.connexdds.PersistentSynchronizationKind* attribute), 414
- not_alive_count (*rti.connexdds.LivelinessChangedStatus* property), 328
- not_alive_count_change (*rti.connexdds.LivelinessChangedStatus* property), 328
- NOT_ALIVE_DISPOSED (*rti.connexdds.InstanceState* attribute), 311
- NOT_ALIVE_FIRST (*rti.connexdds.IgnoredEntityReplacementKind* attribute), 307
- NOT_ALIVE_FIRST (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind* attribute), 306
- NOT_ALIVE_MASK (*rti.connexdds.InstanceState* attribute), 311
- NOT_ALIVE_NO_WRITERS (*rti.connexdds.InstanceState* attribute), 311
- NOT_LOST (*rti.connexdds.SampleLostState* attribute), 510
- NOT_NEW_VIEW (*rti.connexdds.ViewState* attribute), 744
- NOT_READ (*rti.connexdds.SampleState* attribute), 516
- NOT_REJECTED (*rti.connexdds.SampleRejectedState* attribute), 513
- NOT_SET (*rti.connexdds.CdrPaddingKind* attribute), 62

- NOT_SET (*rti.connexdds.CdrPaddingKind.CdrPaddingKind attribute*), 61
 NotAllowedBySecurityError, 365
 NotEnabledError, 365
 NOTICE (*rti.connexdds.SyslogVerbosity attribute*), 629
 NOTICE (*rti.logging.distlog.LogLevel attribute*), 771
 notice () (*rti.connexdds.Logger method*), 344
 notice () (*rti.logging.distlog.Logger static method*), 772
 notify_datareaders () (*rti.connexdds.Subscriber method*), 580
 NullReferenceError, 365
- O**
- OctetSeq (*in module rti.connexdds*), 366
 OctetType (*in module rti.connexdds*), 366
 OFF (*rti.connexdds.PersistentJournalKind attribute*), 412
 OFF (*rti.connexdds.PersistentSynchronizationKind attribute*), 414
 OFFERED_DEADLINE_MISSED (*rti.connexdds.StatusMask attribute*), 566
 offered_deadline_missed_status (*rti.connexdds.Data Writer property*), 117
 offered_deadline_missed_status (*rti.connexdds.DynamicData.Data Writer property*), 201
 offered_deadline_missed_status (*rti.connexdds.ParticipantBuiltinTopicData.Data Writer property*), 387
 offered_deadline_missed_status (*rti.connexdds.PublicationBuiltinTopicData.Data Writer property*), 443
 offered_deadline_missed_status (*rti.connexdds.ServiceRequest.Data Writer property*), 540
 offered_deadline_missed_status (*rti.connexdds.SubscriptionBuiltinTopicData.Data Writer property*), 602
 offered_deadline_missed_status (*rti.connexdds.TopicBuiltinTopicData.Data Writer property*), 657
 OFFERED_INCOMPATIBLE_QOS (*rti.connexdds.StatusMask attribute*), 566
 offered_incompatible_qos_status (*rti.connexdds.Data Writer property*), 117
 offered_incompatible_qos_status (*rti.connexdds.DynamicData.Data Writer property*), 201
 offered_incompatible_qos_status (*rti.connexdds.ParticipantBuiltinTopicData.Data Writer property*), 387
 offered_incompatible_qos_status (*rti.connexdds.PublicationBuiltinTopicData.Data Writer property*), 443
 offered_incompatible_qos_status (*rti.connexdds.ServiceRequest.Data Writer property*), 540
 offered_incompatible_qos_status (*rti.connexdds.SubscriptionBuiltinTopicData.Data Writer property*), 602
 offered_incompatible_qos_status (*rti.connexdds.TopicBuiltinTopicData.Data Writer property*), 657
 OfferedDeadlineMissedStatus (*class in rti.connexdds*), 366
 OfferedIncompatibleQosStatus (*class in rti.connexdds*), 366
 old_source_timestamp_dropped_sample_count (*rti.connexdds.DataReaderCacheStatus property*), 91
 on_application_acknowledgment () (*rti.connexdds.AnyData WriterListener method*), 45
 on_application_acknowledgment () (*rti.connexdds.Data WriterListener method*), 121
 on_application_acknowledgment () (*rti.connexdds.DynamicData.Data WriterListener method*), 205
 on_application_acknowledgment () (*rti.connexdds.DynamicData.NoOpData WriterListener method*), 212
 on_application_acknowledgment () (*rti.connexdds.NoOpAnyData WriterListener method*), 361
 on_application_acknowledgment () (*rti.connexdds.NoOpData WriterListener method*), 363
 on_application_acknowledgment () (*rti.connexdds.ParticipantBuiltinTopicData.Data WriterListener method*), 391
 on_application_acknowledgment () (*rti.connexdds.ParticipantBuiltinTopicData.NoOpData WriterListener method*), 397
 on_application_acknowledgment () (*rti.connexdds.PublicationBuiltinTopicData.Data WriterListener method*), 447
 on_application_acknowledgment () (*rti.connexdds.PublicationBuiltinTopicData.NoOpData WriterListener method*), 453
 on_application_acknowledgment () (*rti.connexdds.ServiceRequest.Data WriterListener method*), 544
 on_application_acknowledgment () (*rti.connexdds.ServiceRequest.NoOpData WriterListener method*), 549
 on_application_acknowledgment () (*rti.connexdds.SubscriptionBuiltinTopicData.Data WriterListener method*), 606
 on_application_acknowledgment () (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpData WriterListener method*), 612
 on_application_acknowledgment () (*rti.connexdds.TopicBuiltinTopicData.Data WriterListener method*), 661
 on_application_acknowledgment ()

(rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener method), 667
on_data_available() *(rti.connexdds.AnyDataReaderListener method)*, 42
on_data_available() *(rti.connexdds.DataReaderListener method)*, 94
on_data_available() *(rti.connexdds.DynamicData.DataReaderListener method)*, 194
on_data_available() *(rti.connexdds.DynamicData.NoOpDataReaderListener method)*, 211
on_data_available() *(rti.connexdds.NoOpAnyDataReaderListener method)*, 360
on_data_available() *(rti.connexdds.NoOpDataReaderListener method)*, 363
on_data_available() *(rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener method)*, 380
on_data_available() *(rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method)*, 396
on_data_available() *(rti.connexdds.PublicationBuiltinTopicData.DataReaderListener method)*, 436
on_data_available() *(rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener method)*, 452
on_data_available() *(rti.connexdds.ServiceRequest.DataReaderListener method)*, 533
on_data_available() *(rti.connexdds.ServiceRequest.NoOpDataReaderListener method)*, 549
on_data_available() *(rti.connexdds.SubscriptionBuiltinTopicData.DataReaderListener method)*, 595
on_data_available() *(rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method)*, 611
on_data_available() *(rti.connexdds.TopicBuiltinTopicData.DataReaderListener method)*, 650
on_data_available() *(rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener method)*, 665
on_data_on_readers() *(rti.connexdds.NoOpSubscriberListener method)*, 365
on_data_on_readers() *(rti.connexdds.SubscriberListener method)*, 580
ON_DEMAND_NAME *(rti.connexdds.FlowController attribute)*, 280
on_inconsistent_topic() *(rti.connexdds.AnyTopicListener method)*, 48
on_inconsistent_topic() *(rti.connexdds.DynamicData.NoOpTopicListener method)*, 213
on_inconsistent_topic() *(rti.connexdds.DynamicData.TopicListener method)*, 216
on_inconsistent_topic() *(rti.connexdds.NoOpAnyTopicListener method)*, 362
on_inconsistent_topic() *(rti.connexdds.NoOpTopicListener method)*, 365
on_inconsistent_topic() *(rti.connexdds.ParticipantBuiltinTopicData.NoOpTopicListener method)*, 398
on_inconsistent_topic() *(rti.connexdds.ParticipantBuiltinTopicData.TopicListener method)*, 401
on_inconsistent_topic() *(rti.connexdds.PublicationBuiltinTopicData.NoOpTopicListener method)*, 455
on_inconsistent_topic() *(rti.connexdds.PublicationBuiltinTopicData.TopicListener method)*, 458
on_inconsistent_topic() *(rti.connexdds.ServiceRequest.NoOpTopicListener method)*, 551
on_inconsistent_topic() *(rti.connexdds.ServiceRequest.TopicListener method)*, 553
on_inconsistent_topic() *(rti.connexdds.SubscriptionBuiltinTopicData.NoOpTopicListener method)*, 613
on_inconsistent_topic() *(rti.connexdds.SubscriptionBuiltinTopicData.TopicListener method)*, 616
on_inconsistent_topic() *(rti.connexdds.TopicBuiltinTopicData.NoOpTopicListener method)*, 668
on_inconsistent_topic() *(rti.connexdds.TopicBuiltinTopicData.TopicListener method)*, 671
on_inconsistent_topic() *(rti.connexdds.TopicListener method)*, 682
on_instance_replaced() *(rti.connexdds.AnyDataWriterListener method)*, 45
on_instance_replaced() *(rti.connexdds.DataWriterListener method)*, 121
on_instance_replaced() *(rti.connexdds.DynamicData.DataWriterListener method)*, 205
on_instance_replaced() *(rti.connexdds.DynamicData.NoOpDataWriterListener method)*, 212
on_instance_replaced() *(rti.connexdds.NoOpAnyDataWriterListener method)*, 361
on_instance_replaced() *(rti.connexdds.NoOpDataWriterListener method)*, 363
on_instance_replaced() *(rti.connexdds.ParticipantBuiltinTopicData.DataWriterListener method)*, 391

- `on_instance_replaced()` (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 397
`on_instance_replaced()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterListener method*), 447
`on_instance_replaced()` (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 454
`on_instance_replaced()` (*rti.connexdds.ServiceRequest.DataWriterListener method*), 544
`on_instance_replaced()` (*rti.connexdds.ServiceRequest.NoOpDataWriterListener method*), 550
`on_instance_replaced()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 606
`on_instance_replaced()` (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 612
`on_instance_replaced()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterListener method*), 661
`on_instance_replaced()` (*rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 667
`on_invalid_local_identity_status_advance_notice()` (*rti.connexdds.DomainParticipantListener method*), 166
`on_invalid_local_identity_status_advance_notice()` (*rti.connexdds.NoOpDomainParticipantListener method*), 364
`on_liveliness_changed()` (*rti.connexdds.AnyDataReaderListener method*), 42
`on_liveliness_changed()` (*rti.connexdds.DataReaderListener method*), 94
`on_liveliness_changed()` (*rti.connexdds.DynamicData.DataReaderListener method*), 194
`on_liveliness_changed()` (*rti.connexdds.DynamicData.NoOpDataReaderListener method*), 211
`on_liveliness_changed()` (*rti.connexdds.NoOpAnyDataReaderListener method*), 360
`on_liveliness_changed()` (*rti.connexdds.NoOpDataReaderListener method*), 363
`on_liveliness_changed()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener method*), 380
`on_liveliness_changed()` (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 396
`on_liveliness_changed()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderListener method*), 436
`on_liveliness_changed()` (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 452
`on_liveliness_changed()` (*rti.connexdds.ServiceRequest.DataReaderListener method*), 533
`on_liveliness_changed()` (*rti.connexdds.ServiceRequest.NoOpDataReaderListener method*), 549
`on_liveliness_changed()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 595
`on_liveliness_changed()` (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 611
`on_liveliness_changed()` (*rti.connexdds.TopicBuiltinTopicData.DataReaderListener method*), 650
`on_liveliness_changed()` (*rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 666
`on_liveliness_lost()` (*rti.connexdds.AnyDataWriterListener method*), 45
`on_liveliness_lost()` (*rti.connexdds.DataWriterListener method*), 121
`on_liveliness_lost()` (*rti.connexdds.DynamicData.DataWriterListener method*), 205
`on_liveliness_lost()` (*rti.connexdds.DynamicData.NoOpDataWriterListener method*), 212
`on_liveliness_lost()` (*rti.connexdds.NoOpAnyDataWriterListener method*), 361
`on_liveliness_lost()` (*rti.connexdds.NoOpDataWriterListener method*), 364
`on_liveliness_lost()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterListener method*), 391
`on_liveliness_lost()` (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 397
`on_liveliness_lost()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterListener method*), 448
`on_liveliness_lost()` (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 454
`on_liveliness_lost()` (*rti.connexdds.ServiceRequest.DataWriterListener method*), 544
`on_liveliness_lost()` (*rti.connexdds.ServiceRequest.NoOpDataWriterListener method*), 550
`on_liveliness_lost()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 606

- `on_liveliness_lost()` (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener* method), 612
- `on_liveliness_lost()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterListener* method), 661
- `on_liveliness_lost()` (*rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener* method), 667
- `on_offered_deadline_missed()` (*rti.connexdds.AnyDataWriterListener* method), 45
- `on_offered_deadline_missed()` (*rti.connexdds.DataWriterListener* method), 121
- `on_offered_deadline_missed()` (*rti.connexdds.DynamicData.DataWriterListener* method), 205
- `on_offered_deadline_missed()` (*rti.connexdds.DynamicData.NoOpDataWriterListener* method), 212
- `on_offered_deadline_missed()` (*rti.connexdds.NoOpAnyDataWriterListener* method), 361
- `on_offered_deadline_missed()` (*rti.connexdds.NoOpDataWriterListener* method), 364
- `on_offered_deadline_missed()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterListener* method), 391
- `on_offered_deadline_missed()` (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataWriterListener* method), 397
- `on_offered_deadline_missed()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterListener* method), 448
- `on_offered_deadline_missed()` (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener* method), 454
- `on_offered_deadline_missed()` (*rti.connexdds.ServiceRequest.DataWriterListener* method), 544
- `on_offered_deadline_missed()` (*rti.connexdds.ServiceRequest.NoOpDataWriterListener* method), 550
- `on_offered_deadline_missed()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener* method), 606
- `on_offered_deadline_missed()` (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener* method), 612
- `on_offered_deadline_missed()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterListener* method), 661
- `on_offered_deadline_missed()` (*rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener* method), 667
- `on_offered_incompatible_qos()` (*rti.connexdds.AnyDataWriterListener* method), 45
- `on_offered_incompatible_qos()` (*rti.connexdds.DataWriterListener* method), 121
- `on_offered_incompatible_qos()` (*rti.connexdds.DynamicData.DataWriterListener* method), 205
- `on_offered_incompatible_qos()` (*rti.connexdds.DynamicData.NoOpDataWriterListener* method), 212
- `on_offered_incompatible_qos()` (*rti.connexdds.NoOpAnyDataWriterListener* method), 362
- `on_offered_incompatible_qos()` (*rti.connexdds.NoOpDataWriterListener* method), 364
- `on_offered_incompatible_qos()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterListener* method), 391
- `on_offered_incompatible_qos()` (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataWriterListener* method), 398
- `on_offered_incompatible_qos()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterListener* method), 448
- `on_offered_incompatible_qos()` (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener* method), 454
- `on_offered_incompatible_qos()` (*rti.connexdds.ServiceRequest.DataWriterListener* method), 544
- `on_offered_incompatible_qos()` (*rti.connexdds.ServiceRequest.NoOpDataWriterListener* method), 550
- `on_offered_incompatible_qos()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener* method), 606
- `on_offered_incompatible_qos()` (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener* method), 612
- `on_offered_incompatible_qos()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterListener* method), 661
- `on_offered_incompatible_qos()` (*rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener* method), 667
- `on_publication_matched()` (*rti.connexdds.AnyDataWriterListener* method), 46
- `on_publication_matched()` (*rti.connexdds.DataWriterListener* method), 121
- `on_publication_matched()` (*rti.connexdds.DynamicData.DataWriterListener* method), 205
- `on_publication_matched()` (*rti.connexdds.DynamicData.NoOpDataWriterListener* method), 213
- `on_publication_matched()` (*rti.connexdds.NoOpAnyDataWriterListener* method), 362
- `on_publication_matched()` (*rti.connexdds.NoOpDataWriterListener* method), 364
- `on_publication_matched()` (*rti.connexdds.Participant*

BuiltinTopicData.DataWriterListener method),
 392

on_publication_matched() (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*),
 398

on_publication_matched() (*rti.connexdds.PublicationBuiltinTopicData.DataWriterListener method*),
 448

on_publication_matched() (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*),
 454

on_publication_matched()
 (*rti.connexdds.ServiceRequest.DataWriterListener method*), 544

on_publication_matched() (*rti.connexdds.ServiceRequest.NoOpDataWriterListener method*),
 550

on_publication_matched() (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener method*),
 606

on_publication_matched() (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 613

on_publication_matched() (*rti.connexdds.TopicBuiltinTopicData.DataWriterListener method*),
 661

on_publication_matched() (*rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener method*),
 667

on_reliable_reader_activity_changed()
 (*rti.connexdds.AnyDataWriterListener method*), 46

on_reliable_reader_activity_changed()
 (*rti.connexdds.DataWriterListener method*), 121

on_reliable_reader_activity_changed()
 (*rti.connexdds.DynamicData.DataWriterListener method*), 205

on_reliable_reader_activity_changed()
 (*rti.connexdds.DynamicData.NoOpDataWriterListener method*),
 213

on_reliable_reader_activity_changed()
 (*rti.connexdds.NoOpAnyDataWriterListener method*),
 362

on_reliable_reader_activity_changed()
 (*rti.connexdds.NoOpDataWriterListener method*),
 364

on_reliable_reader_activity_changed()
 (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterListener method*),
 392

on_reliable_reader_activity_changed()
 (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*),
 398

on_reliable_reader_activity_changed()
 (*rti.connexdds.PublicationBuiltinTopicData.DataWriterListener method*),
 448

on_reliable_reader_activity_changed()
 (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*),
 454

on_reliable_reader_activity_changed()
 (*rti.connexdds.ServiceRequest.DataWriterListener method*), 544

on_reliable_reader_activity_changed()
 (*rti.connexdds.ServiceRequest.NoOpDataWriterListener method*),
 550

on_reliable_reader_activity_changed()
 (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener method*),
 607

on_reliable_reader_activity_changed()
 (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*),
 613

on_reliable_reader_activity_changed()
 (*rti.connexdds.TopicBuiltinTopicData.DataWriterListener method*),
 661

on_reliable_reader_activity_changed()
 (*rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener method*),
 667

on_reliable_writer_cache_changed()
 (*rti.connexdds.AnyDataWriterListener method*), 46

on_reliable_writer_cache_changed()
 (*rti.connexdds.DataWriterListener method*), 121

on_reliable_writer_cache_changed()
 (*rti.connexdds.DynamicData.DataWriterListener method*), 206

on_reliable_writer_cache_changed() (*rti.connexdds.DynamicData.NoOpDataWriterListener method*), 213

on_reliable_writer_cache_changed()
 (*rti.connexdds.NoOpAnyDataWriterListener method*),
 362

on_reliable_writer_cache_changed()
 (*rti.connexdds.NoOpDataWriterListener method*),
 364

on_reliable_writer_cache_changed()
 (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterListener method*),
 392

on_reliable_writer_cache_changed()
 (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*),
 398

on_reliable_writer_cache_changed()
 (*rti.connexdds.PublicationBuiltinTopicData.DataWriterListener method*),
 448

on_reliable_writer_cache_changed()
 (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*),
 454

on_reliable_writer_cache_changed()

- (*rti.connexdds.ServiceRequest.DataWriterListener method*), 545
- on_reliable_writer_cache_changed()* (*rti.connexdds.ServiceRequest.NoOpDataWriterListener method*), 550
- on_reliable_writer_cache_changed()* (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 607
- on_reliable_writer_cache_changed()* (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 613
- on_reliable_writer_cache_changed()* (*rti.connexdds.TopicBuiltinTopicData.DataWriterListener method*), 662
- on_reliable_writer_cache_changed()* (*rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 667
- on_reply_available* (*rti.rpc.Requester property*), 762
- on_request_available* (*rti.rpc.Replier property*), 765
- on_requested_deadline_missed()* (*rti.connexdds.AnyDataReaderListener method*), 42
- on_requested_deadline_missed()* (*rti.connexdds.DataReaderListener method*), 94
- on_requested_deadline_missed()* (*rti.connexdds.DynamicData.DataReaderListener method*), 194
- on_requested_deadline_missed()* (*rti.connexdds.DynamicData.NoOpDataReaderListener method*), 211
- on_requested_deadline_missed()* (*rti.connexdds.NoOpAnyDataReaderListener method*), 361
- on_requested_deadline_missed()* (*rti.connexdds.NoOpDataReaderListener method*), 363
- on_requested_deadline_missed()* (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener method*), 380
- on_requested_deadline_missed()* (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 396
- on_requested_deadline_missed()* (*rti.connexdds.PublicationBuiltinTopicData.DataReaderListener method*), 436
- on_requested_deadline_missed()* (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 453
- on_requested_deadline_missed()* (*rti.connexdds.ServiceRequest.DataReaderListener method*), 533
- on_requested_deadline_missed()* (*rti.connexdds.ServiceRequest.NoOpDataReaderListener method*), 549
- on_requested_deadline_missed()* (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 595
- on_requested_deadline_missed()* (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 611
- on_requested_deadline_missed()* (*rti.connexdds.TopicBuiltinTopicData.DataReaderListener method*), 650
- on_requested_deadline_missed()* (*rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 666
- on_requested_incompatible_qos()* (*rti.connexdds.AnyDataReaderListener method*), 42
- on_requested_incompatible_qos()* (*rti.connexdds.DataReaderListener method*), 94
- on_requested_incompatible_qos()* (*rti.connexdds.DynamicData.DataReaderListener method*), 194
- on_requested_incompatible_qos()* (*rti.connexdds.DynamicData.NoOpDataReaderListener method*), 211
- on_requested_incompatible_qos()* (*rti.connexdds.NoOpAnyDataReaderListener method*), 361
- on_requested_incompatible_qos()* (*rti.connexdds.NoOpDataReaderListener method*), 363
- on_requested_incompatible_qos()* (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener method*), 380
- on_requested_incompatible_qos()* (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 396
- on_requested_incompatible_qos()* (*rti.connexdds.PublicationBuiltinTopicData.DataReaderListener method*), 436
- on_requested_incompatible_qos()* (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 453
- on_requested_incompatible_qos()* (*rti.connexdds.ServiceRequest.DataReaderListener method*), 533
- on_requested_incompatible_qos()* (*rti.connexdds.ServiceRequest.NoOpDataReaderListener method*), 549
- on_requested_incompatible_qos()* (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 595
- on_requested_incompatible_qos()*

(rti.connexdds.SubscriptionBuiltinTopicData.NoOp-DataReaderListener method), 611
 on_requested_incompatible_qos() (*rti.connexdds.TopicBuiltinTopicData.DataReaderListener method*), 650
 on_requested_incompatible_qos() (*rti.connexdds.TopicBuiltinTopicData.NoOp-DataReaderListener method*), 666
 on_sample_lost() (*rti.connexdds.AnyDataReaderListener method*), 42
 on_sample_lost() (*rti.connexdds.DataReaderListener method*), 94
 on_sample_lost() (*rti.connexdds.DynamicData.DataReaderListener method*), 194
 on_sample_lost() (*rti.connexdds.DynamicData.NoOp-DataReaderListener method*), 211
 on_sample_lost() (*rti.connexdds.NoOpAnyDataReaderListener method*), 361
 on_sample_lost() (*rti.connexdds.NoOpDataReaderListener method*), 363
 on_sample_lost() (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener method*), 380
 on_sample_lost() (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 396
 on_sample_lost() (*rti.connexdds.PublicationBuiltinTopicData.DataReaderListener method*), 437
 on_sample_lost() (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 453
 on_sample_lost() (*rti.connexdds.ServiceRequest.DataReaderListener method*), 534
 on_sample_lost() (*rti.connexdds.ServiceRequest.NoOp-DataReaderListener method*), 549
 on_sample_lost() (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 595
 on_sample_lost() (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 611
 on_sample_lost() (*rti.connexdds.TopicBuiltinTopicData.DataReaderListener method*), 650
 on_sample_lost() (*rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 666
 on_sample_rejected() (*rti.connexdds.AnyDataReaderListener method*), 42
 on_sample_rejected() (*rti.connexdds.DataReaderListener method*), 94
 on_sample_rejected() (*rti.connexdds.DynamicData.DataReaderListener method*), 195
 on_sample_rejected() (*rti.connexdds.DynamicData.NoOpDataReaderListener method*), 212
 on_sample_rejected() (*rti.connexdds.NoOpAnyDataReaderListener method*), 361
 on_sample_rejected() (*rti.connexdds.NoOpDataReaderListener method*), 363
 on_sample_rejected() (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener method*), 380
 on_sample_rejected() (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 397
 on_sample_rejected() (*rti.connexdds.PublicationBuiltinTopicData.DataReaderListener method*), 437
 on_sample_rejected() (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 453
 on_sample_rejected() (*rti.connexdds.ServiceRequest.DataReaderListener method*), 534
 on_sample_rejected() (*rti.connexdds.ServiceRequest.NoOpDataReaderListener method*), 549
 on_sample_rejected() (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 595
 on_sample_rejected() (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 612
 on_sample_rejected() (*rti.connexdds.TopicBuiltinTopicData.DataReaderListener method*), 650
 on_sample_rejected() (*rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 666
 on_service_request_accepted() (*rti.connexdds.AnyDataWriterListener method*), 46
 on_service_request_accepted() (*rti.connexdds.DataWriterListener method*), 122
 on_service_request_accepted() (*rti.connexdds.DynamicData.DataWriterListener method*), 206
 on_service_request_accepted() (*rti.connexdds.DynamicData.NoOpDataWriterListener method*), 213
 on_service_request_accepted() (*rti.connexdds.NoOpAnyDataWriterListener method*), 362
 on_service_request_accepted() (*rti.connexdds.NoOpDataWriterListener method*), 364

- `on_service_request_accepted()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterListener method*), 392
- `on_service_request_accepted()` (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataWriterListener method*), 398
- `on_service_request_accepted()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriterListener method*), 449
- `on_service_request_accepted()` (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataWriterListener method*), 455
- `on_service_request_accepted()` (*rti.connexdds.ServiceRequest.DataWriterListener method*), 545
- `on_service_request_accepted()` (*rti.connexdds.ServiceRequest.NoOpDataWriterListener method*), 551
- `on_service_request_accepted()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterListener method*), 607
- `on_service_request_accepted()` (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataWriterListener method*), 613
- `on_service_request_accepted()` (*rti.connexdds.TopicBuiltinTopicData.DataWriterListener method*), 662
- `on_service_request_accepted()` (*rti.connexdds.TopicBuiltinTopicData.NoOpDataWriterListener method*), 668
- `on_subscription_matched()` (*rti.connexdds.AnyDataReaderListener method*), 43
- `on_subscription_matched()` (*rti.connexdds.DataReaderListener method*), 95
- `on_subscription_matched()` (*rti.connexdds.DynamicData.DataReaderListener method*), 195
- `on_subscription_matched()` (*rti.connexdds.DynamicData.NoOpDataReaderListener method*), 212
- `on_subscription_matched()` (*rti.connexdds.NoOpAnyDataReaderListener method*), 361
- `on_subscription_matched()` (*rti.connexdds.NoOpDataReaderListener method*), 363
- `on_subscription_matched()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderListener method*), 381
- `on_subscription_matched()` (*rti.connexdds.ParticipantBuiltinTopicData.NoOpDataReaderListener method*), 397
- `on_subscription_matched()` (*rti.connexdds.PublicationBuiltinTopicData.DataReaderListener method*), 437
- `on_subscription_matched()` (*rti.connexdds.PublicationBuiltinTopicData.NoOpDataReaderListener method*), 453
- `on_subscription_matched()` (*rti.connexdds.ServiceRequest.DataReaderListener method*), 534
- `on_subscription_matched()` (*rti.connexdds.ServiceRequest.NoOpDataReaderListener method*), 549
- `on_subscription_matched()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderListener method*), 595
- `on_subscription_matched()` (*rti.connexdds.SubscriptionBuiltinTopicData.NoOpDataReaderListener method*), 612
- `on_subscription_matched()` (*rti.connexdds.TopicBuiltinTopicData.DataReaderListener method*), 650
- `on_subscription_matched()` (*rti.connexdds.TopicBuiltinTopicData.NoOpDataReaderListener method*), 666
- `operation()` (*in module rti.rpc*), 769
- `optional` (*rti.connexdds.Member property*), 347
- `ordered_access` (*rti.connexdds.Presentation property*), 415
- `ordinal` (*rti.connexdds.EnumMember property*), 263
- `original_publication_virtual_guid` (*rti.connexdds.SampleInfo property*), 508
- `original_publication_virtual_sample_identity` (*rti.connexdds.SampleInfo property*), 508
- `original_publication_virtual_sequence_number` (*rti.connexdds.SampleInfo property*), 509
- `original_related_reader_guid` (*rti.connexdds.TopicQueryData property*), 687
- `out_of_range_rejected_sample_count` (*rti.connexdds.DataReaderProtocolStatus property*), 96
- `OutOfResourcesError`, 366
- `output_file()` (*rti.connexdds.Logger method*), 344
- `output_file_set()` (*rti.connexdds.Logger method*), 344
- `output_handler()` (*rti.connexdds.Logger method*), 344
- `outstanding_asynchronous_sample_allocation` (*rti.connexdds.DomainParticipantResourceLimits property*), 173
- `Ownership` (*class in rti.connexdds*), 366
- `ownership` (*rti.connexdds.DataReaderQos property*), 103
- `ownership` (*rti.connexdds.DataWriterQos property*), 131
- `ownership` (*rti.connexdds.PublicationBuiltinTopicData property*), 462
- `ownership` (*rti.connexdds.SubscriptionBuiltinTopicData property*), 621
- `ownership` (*rti.connexdds.TopicBuiltinTopicData property*), 675
- `ownership` (*rti.connexdds.TopicQos property*), 685
- `ownership_dropped_sample_count` (*rti.connexdds.DataReaderCacheStatus property*), 91
- `ownership_strength` (*rti.connexdds.DataWriterQos property*), 131
- `ownership_strength` (*rti.connexdds.PublicationBuiltinTopicData property*), 462
- `OwnershipKind` (*class in rti.connexdds*), 367

OwnershipKind.OwnershipKind (class in *rti.connexdds*), 367

OwnershipStrength (class in *rti.connexdds*), 369

P

parameter_count (*rti.connexdds.Filter* property), 271

parameters (*rti.connexdds.Query* property), 485

parameters (*rti.connexdds.QueryCondition* property), 486

parameters_length (*rti.connexdds.Query* property), 485

parameters_length (*rti.connexdds.QueryCondition* property), 486

parent (*rti.connexdds.StructType* property), 578

partial_configuration (*rti.connexdds.ParticipantBuiltinTopicData* property), 405

participant (*rti.connexdds.DynamicData.ITopicDescription* property), 209

participant (*rti.connexdds.FlowController* property), 280

participant (*rti.connexdds.ITopicDescription* property), 305

participant (*rti.connexdds.ParticipantBuiltinTopicData.ITopicDescription* property), 395

participant (*rti.connexdds.PublicationBuiltinTopicData.ITopicDescription* property), 451

participant (*rti.connexdds.Publisher* property), 473

participant (*rti.connexdds.ServiceRequest.ITopicDescription* property), 547

participant (*rti.connexdds.Subscriber* property), 580

participant (*rti.connexdds.SubscriptionBuiltinTopicData.ITopicDescription* property), 610

participant (*rti.connexdds.TopicBuiltinTopicData.ITopicDescription* property), 664

participant (*rti.logging.distlog.LoggerOptions* property), 772

participant_announcement_period (*rti.connexdds.DiscoveryConfig* property), 149

participant_factory_qos (*rti.connexdds.DomainParticipant* attribute), 162

participant_id (*rti.connexdds.WireProtocol* property), 752

participant_id_gain (*rti.connexdds.RtpsWellKnownPorts* property), 504

participant_key (*rti.connexdds.PublicationBuiltinTopicData* property), 463

participant_key (*rti.connexdds.SubscriptionBuiltinTopicData* property), 621

participant_liveliness_assert_period (*rti.connexdds.DiscoveryConfig* property), 149

participant_liveliness_lease_duration (*rti.connexdds.DiscoveryConfig* property), 149

participant_message_reader (*rti.connexdds.DiscoveryConfig* property), 149

participant_message_reader_reliability_kind (*rti.connexdds.DiscoveryConfig* property), 149

participant_message_writer (*rti.connexdds.DiscoveryConfig* property), 149

participant_name (*rti.connexdds.DomainParticipantConfigParams* property), 164

participant_name (*rti.connexdds.DomainParticipantQos* property), 170

participant_name (*rti.connexdds.ParticipantBuiltinTopicData* property), 405

participant_property_list_max_length (*rti.connexdds.DomainParticipantResourceLimits* property), 173

participant_property_string_max_length (*rti.connexdds.DomainParticipantResourceLimits* property), 173

participant_protocol_status (*rti.connexdds.DomainParticipant* property), 162

participant_qos (*rti.connexdds.QosProvider* property), 483

participant_qos_from_profile() (*rti.connexdds.QosProvider* method), 483

participant_qos_library_name (*rti.connexdds.DomainParticipantConfigParams* property), 164

participant_qos_profile_name (*rti.connexdds.DomainParticipantConfigParams* property), 164

participant_qos_profile_name (*rti.connexdds.MonitoringDedicatedParticipantSettings* property), 350

participant_reader (*rti.connexdds.DomainParticipant* property), 162

participant_reader_resource_limits (*rti.connexdds.DiscoveryConfig* property), 149

participant_user_data_max_length (*rti.connexdds.DomainParticipantResourceLimits* property), 174

ParticipantBuiltinTopicData (class in *rti.connexdds*), 369

ParticipantBuiltinTopicData.ContentFilter (class in *rti.connexdds*), 369

ParticipantBuiltinTopicData.ContentFilteredTopic (class in *rti.connexdds*), 370

ParticipantBuiltinTopicData.ContentFilteredTopicSeq (class in *rti.connexdds*), 371

ParticipantBuiltinTopicData.DataReader (class in *rti.connexdds*), 373

ParticipantBuiltinTopicData.DataReaderListener (class in *rti.connexdds*), 380

ParticipantBuiltinTopicData.DataReader.Selector (class in *rti.connexdds*), 373

ParticipantBuiltinTopicData.DataReaderSeq (class in *rti.connexdds*), 381

ParticipantBuiltinTopicData.DataWriter (class

- in rti.connexdds*), 383
- ParticipantBuiltinTopicData.DataWriterListener (class in *rti.connexdds*), 391
- ParticipantBuiltinTopicData.DataWriterSeq (class in *rti.connexdds*), 392
- ParticipantBuiltinTopicData.ITopicDescription (class in *rti.connexdds*), 394
- ParticipantBuiltinTopicData.LoanedSample (class in *rti.connexdds*), 395
- ParticipantBuiltinTopicData.LoanedSamples (class in *rti.connexdds*), 395
- ParticipantBuiltinTopicData.NoOpDataReaderListener (class in *rti.connexdds*), 396
- ParticipantBuiltinTopicData.NoOp-DataWriterListener (class in *rti.connexdds*), 397
- ParticipantBuiltinTopicData.NoOpTopicListener (class in *rti.connexdds*), 398
- ParticipantBuiltinTopicData.Sample (class in *rti.connexdds*), 399
- ParticipantBuiltinTopicDataSeq (class in *rti.connexdds*), 406
- ParticipantBuiltinTopicData.SharedSamples (class in *rti.connexdds*), 399
- ParticipantBuiltinTopicData.TimestampedSeq (class in *rti.connexdds*), 408
- ParticipantBuiltinTopicData.Topic (class in *rti.connexdds*), 400
- ParticipantBuiltinTopicData.TopicDescription (class in *rti.connexdds*), 400
- ParticipantBuiltinTopicData.TopicListener (class in *rti.connexdds*), 401
- ParticipantBuiltinTopicData.TopicSeq (class in *rti.connexdds*), 401
- ParticipantBuiltinTopicData.ValidLoanedSamples (class in *rti.connexdds*), 403
- ParticipantBuiltinTopicData.WriterContentFilter (class in *rti.connexdds*), 404
- ParticipantBuiltinTopicData.WriterContentFilterHelper (class in *rti.connexdds*), 405
- Partition (class in *rti.connexdds*), 411
- partition (*rti.connexdds.DomainParticipantQos* property), 170
- partition (*rti.connexdds.PublicationBuiltinTopicData* property), 463
- partition (*rti.connexdds.PublisherQos* property), 476
- partition (*rti.connexdds.SubscriberQos* property), 582
- partition (*rti.connexdds.SubscriptionBuiltinTopicData* property), 621
- pattern_alarm_event (*rti.connexdds.BuiltinProfiles* attribute), 58
- pattern_alarm_status (*rti.connexdds.BuiltinProfiles* attribute), 58
- pattern_event (*rti.connexdds.BuiltinProfiles* attribute), 58
- pattern_last_value_cache (*rti.connexdds.BuiltinProfiles* attribute), 58
- pattern_periodic_data (*rti.connexdds.BuiltinProfiles* attribute), 58
- pattern_reliable_streaming (*rti.connexdds.BuiltinProfiles* attribute), 58
- pattern_status (*rti.connexdds.BuiltinProfiles* attribute), 58
- pattern_streaming (*rti.connexdds.BuiltinProfiles* attribute), 58
- period (*rti.connexdds.Deadline* property), 142
- period (*rti.connexdds.FlowControllerTokenBucketProperty* property), 285
- periodic_settings (*rti.connexdds.MonitoringDistributionSettings* property), 351
- PERSIST (*rti.connexdds.PersistentJournalKind* attribute), 412
- PERSISTENCE (*rti.connexdds.ServiceKind* attribute), 521
- PERSISTENCE (*rti.connexdds.ServiceKind.ServiceKind* attribute), 522
- persistent (*rti.connexdds.Durability* attribute), 178
- PERSISTENT (*rti.connexdds.DurabilityKind* attribute), 180
- PERSISTENT (*rti.connexdds.DurabilityKind.DurabilityKind* attribute), 179
- PersistentJournalKind (class in *rti.connexdds*), 412
- PersistentStorageSettings (class in *rti.connexdds*), 413
- PersistentSynchronizationKind (class in *rti.connexdds*), 414
- PL_CDR_BIG_ENDIAN (*rti.connexdds.DynamicDataEncapsulationKind* attribute), 246
- PL_CDR_BIG_ENDIAN (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind* attribute), 245
- PL_CDR_LITTLE_ENDIAN (*rti.connexdds.DynamicDataEncapsulationKind* attribute), 246
- PL_CDR_LITTLE_ENDIAN (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind* attribute), 245
- platform (*rti.connexdds.LogCategory* attribute), 344
- platform (*rti.connexdds.LogCategory.LogCategory* attribute), 342
- pointer (*rti.connexdds.Member* property), 347
- pointer (*rti.connexdds.UnionMember* property), 738
- policies (*rti.connexdds.OfferedIncompatibleQosStatus* property), 366
- policies (*rti.connexdds.RequestedIncompatibleQosStatus* property), 496
- policy (*rti.connexdds.QosPolicyCount* property), 478
- polling_period (*rti.connexdds.MonitoringPeriodicDistributionSettings* property), 356
- pop () (*rti.connexdds.AnyDataReaderSeq* method), 45
- pop () (*rti.connexdds.AnyDataWriterSeq* method), 48
- pop () (*rti.connexdds.AnyTopicSeq* method), 50

- pop () (*rti.connexdds.BoolSeq method*), 55
- pop () (*rti.connexdds.ChannelSettingsSeq method*), 65
- pop () (*rti.connexdds.CharSeq method*), 67
- pop () (*rti.connexdds.ConditionSeq method*), 75
- pop () (*rti.connexdds.ContentFilteredTopicSeq method*), 79
- pop () (*rti.connexdds.CookieSeq method*), 82
- pop () (*rti.connexdds.DataReaderSeq method*), 109
- pop () (*rti.connexdds.DataWriterSeq method*), 139
- pop () (*rti.connexdds.DomainParticipantSeq method*), 177
- pop () (*rti.connexdds.DynamicData.ContentFilteredTopicSeq method*), 188
- pop () (*rti.connexdds.DynamicData.DataReaderSeq method*), 197
- pop () (*rti.connexdds.DynamicData.DataWriterSeq method*), 208
- pop () (*rti.connexdds.DynamicDataSeq method*), 251
- pop () (*rti.connexdds.DynamicDataTimestampedSeq method*), 253
- pop () (*rti.connexdds.DynamicData.TopicSeq method*), 218
- pop () (*rti.connexdds.EndpointGroupSeq method*), 258
- pop () (*rti.connexdds.EntitySeq method*), 262
- pop () (*rti.connexdds.EnumMemberSeq method*), 265
- pop () (*rti.connexdds.Float32Seq method*), 276
- pop () (*rti.connexdds.Float64Seq method*), 279
- pop () (*rti.connexdds.Float128Seq method*), 274
- pop () (*rti.connexdds.IAnyDataReaderSeq method*), 293
- pop () (*rti.connexdds.IAnyDataWriterSeq method*), 296
- pop () (*rti.connexdds.IAnyTopicSeq method*), 299
- pop () (*rti.connexdds.IConditionSeq method*), 301
- pop () (*rti.connexdds.IEntitySeq method*), 304
- pop () (*rti.connexdds.InstanceHandleSeq method*), 311
- pop () (*rti.connexdds.Int8Seq method*), 325
- pop () (*rti.connexdds.Int16Seq method*), 316
- pop () (*rti.connexdds.Int32Seq method*), 319
- pop () (*rti.connexdds.Int64Seq method*), 322
- pop () (*rti.connexdds.LocatorFilterElementSeq method*), 335
- pop () (*rti.connexdds.LocatorSeq method*), 340
- pop () (*rti.connexdds.MemberSeq method*), 349
- pop () (*rti.connexdds.MonitoringMetricSelectionSeq method*), 355
- pop () (*rti.connexdds.MulticastMappingSeq method*), 360
- pop () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopicSeq method*), 373
- pop () (*rti.connexdds.ParticipantBuiltinTopicData.DataReaderSeq method*), 383
- pop () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriterSeq method*), 394
- pop () (*rti.connexdds.ParticipantBuiltinTopicDataSeq method*), 408
- pop () (*rti.connexdds.ParticipantBuiltinTopicDataTimestampedSeq method*), 411
- pop () (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq method*), 403
- pop () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopicSeq method*), 429
- pop () (*rti.connexdds.PublicationBuiltinTopicData.DataReaderSeq method*), 439
- pop () (*rti.connexdds.PublicationBuiltinTopicData.DataWriterSeq method*), 451
- pop () (*rti.connexdds.PublicationBuiltinTopicDataSeq method*), 466
- pop () (*rti.connexdds.PublicationBuiltinTopicDataTimestampedSeq method*), 468
- pop () (*rti.connexdds.PublicationBuiltinTopicData.TopicSeq method*), 460
- pop () (*rti.connexdds.PublisherSeq method*), 478
- pop () (*rti.connexdds.QosPolicyCountSeq method*), 480
- pop () (*rti.connexdds.ServiceRequest.ContentFilteredTopicSeq method*), 527
- pop () (*rti.connexdds.ServiceRequest.DataReaderSeq method*), 536
- pop () (*rti.connexdds.ServiceRequest.DataWriterSeq method*), 547
- pop () (*rti.connexdds.ServiceRequestSeq method*), 562
- pop () (*rti.connexdds.ServiceRequestTimestampedSeq method*), 564
- pop () (*rti.connexdds.ServiceRequest.TopicSeq method*), 555
- pop () (*rti.connexdds.StringPairSeq method*), 574
- pop () (*rti.connexdds.StringSeq method*), 576
- pop () (*rti.connexdds.SubscriberSeq method*), 584
- pop () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 588
- pop () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 597
- pop () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 609
- pop () (*rti.connexdds.SubscriptionBuiltinTopicDataSeq method*), 624
- pop () (*rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 627
- pop () (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq method*), 618
- pop () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 643
- pop () (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq method*), 652
- pop () (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method*), 664
- pop () (*rti.connexdds.TopicBuiltinTopicDataSeq method*), 678
- pop () (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq method*), 680
- pop () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq method*), 673
- pop () (*rti.connexdds.TopicSeq method*), 692
- pop () (*rti.connexdds.TransportInfoSeq method*), 701

- pop () (*rti.connexdds.TransportMulticastSeq* method), 708
- pop () (*rti.connexdds.TransportMulticastSettingsSeq* method), 711
- pop () (*rti.connexdds.TransportUnicastSettingsSeq* method), 715
- pop () (*rti.connexdds.Uint8Seq* method), 736
- pop () (*rti.connexdds.Uint16Seq* method), 728
- pop () (*rti.connexdds.Uint32Seq* method), 731
- pop () (*rti.connexdds.Uint64Seq* method), 733
- pop () (*rti.connexdds.UnionMemberSeq* method), 740
- pop () (*rti.connexdds.WcharSeq* method), 751
- pop () (*rti.connexdds.WstringSeq* method), 758
- port (*rti.connexdds.Locator* property), 332
- port_base (*rti.connexdds.RtpsWellKnownPorts* property), 504
- PreconditionNotMetError, 415
- Presentation (class in *rti.connexdds*), 415
- presentation (*rti.connexdds.PublicationBuiltinTopicData* property), 463
- presentation (*rti.connexdds.PublisherQos* property), 476
- presentation (*rti.connexdds.SubscriberQos* property), 582
- presentation (*rti.connexdds.SubscriptionBuiltinTopicData* property), 621
- PresentationAccessScopeKind (class in *rti.connexdds*), 416
- PresentationAccessScopeKind.PresentationAccessScopeKind (class in *rti.connexdds*), 416
- pretty_print (*rti.connexdds.PrintFormatProperty* property), 423
- prevent_type_widening (*rti.connexdds.TypeConsistencyEnforcement* property), 717
- PRIMITIVE_TYPE (*rti.connexdds.TypeKind* attribute), 720
- PRIMITIVE_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 722
- print_format (*rti.connexdds.Logger* property), 344
- print_format () (*rti.logging.distlog.Logger* static method), 772
- print_idl () (*rti.connexdds.DynamicType* method), 255
- print_private (*rti.connexdds.QosPrintFormat* property), 481
- PrintFormat (class in *rti.connexdds*), 418
- PrintFormatKind (class in *rti.connexdds*), 420
- PrintFormatKind.PrintFormatKind (class in *rti.connexdds*), 421
- PrintFormat.PrintFormat (class in *rti.connexdds*), 418
- PrintFormatProperty (class in *rti.connexdds*), 422
- priority (*rti.connexdds.ChannelSettings* property), 63
- priority (*rti.connexdds.FilterSampleInfo* property), 272
- priority (*rti.connexdds.PublishMode* property), 470
- priority (*rti.connexdds.ThreadSettings* property), 631
- priority (*rti.connexdds.WriteParams* property), 755
- PRIORITY_ENFORCE (*rti.connexdds.ThreadSettingsKindMask* attribute), 633
- product_version (*rti.connexdds.ParticipantBuiltinTopicData* property), 406
- product_version (*rti.connexdds.PublicationBuiltinTopicData* property), 463
- product_version (*rti.connexdds.SubscriptionBuiltinTopicData* property), 621
- ProductVersion (class in *rti.connexdds*), 423
- profiles_loaded (*rti.connexdds.QosProvider* property), 483
- propagate () (*rti.connexdds.Property* method), 424
- propagate_app_ack_with_no_response (*rti.connexdds.DataWriterProtocol* property), 122
- propagate_dispose_of_unregistered_instances (*rti.connexdds.DataReaderProtocol* property), 95
- propagate_unregister_of_disposed_instances (*rti.connexdds.DataReaderProtocol* property), 95
- Property (class in *rti.connexdds*), 423
- property (*rti.connexdds.DataReaderQos* property), 103
- property (*rti.connexdds.DataWriterQos* property), 131
- property (*rti.connexdds.DomainParticipantQos* property), 170
- property (*rti.connexdds.FlowController* property), 280
- property (*rti.connexdds.ParticipantBuiltinTopicData* property), 406
- property (*rti.connexdds.PublicationBuiltinTopicData* property), 463
- property (*rti.connexdds.SubscriptionBuiltinTopicData* property), 621
- property (*rti.connexdds.WaitSet* property), 748
- PROTOCOL (*rti.connexdds.AcknowledgmentKind* attribute), 36
- PROTOCOL (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind* attribute), 36
- protocol (*rti.connexdds.DataReaderQos* property), 103
- protocol (*rti.connexdds.DataWriterQos* property), 131
- ProtocolVersion (class in *rti.connexdds*), 425
- provider_params (*rti.connexdds.QosProvider* property), 483
- publication_handle (*rti.connexdds.SampleInfo* property), 509
- PUBLICATION_MATCHED (*rti.connexdds.StatusMask* attribute), 566
- publication_matched_status (*rti.connexdds.DataWriter* property), 117
- publication_matched_status (*rti.connexdds.DynamicData.DataWriter* property), 201
- publication_matched_status (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter* property), 387
- publication_matched_status (*rti.connexdds.PublicationBuiltinTopicData.DataWriter* property), 443
- publication_matched_status (*rti.connexdds.ServiceRequest.DataWriter* property), 540
- publication_matched_status (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* property), 602
- publication_matched_status

- (rti.connexdds.TopicBuiltinTopicData.DataWriter property)*, 657
- publication_name *(rti.connexdds.PublicationBuiltinTopicData property)*, 463
- publication_period *(rti.connexdds.MonitoringEventDistributionSettings property)*, 351
- publication_period *(rti.connexdds.MonitoringLoggingDistributionSettings property)*, 352
- publication_period *(rti.connexdds.TopicQueryDispatch property)*, 687
- PUBLICATION_PRIORITY_UNDEFINED *(rti.connexdds.PublishMode attribute)*, 469
- publication_reader *(rti.connexdds.DiscoveryConfig property)*, 149
- publication_reader *(rti.connexdds.DomainParticipant property)*, 162
- publication_reader_resource_limits *(rti.connexdds.DiscoveryConfig property)*, 149
- publication_sequence_number *(rti.connexdds.SampleInfo property)*, 509
- publication_writer *(rti.connexdds.DiscoveryConfig property)*, 149
- publication_writer_data_lifecycle *(rti.connexdds.DiscoveryConfig property)*, 149
- publication_writer_publish_mode *(rti.connexdds.DiscoveryConfig property)*, 150
- PublicationBuiltinTopicData *(class in rti.connexdds)*, 426
- PublicationBuiltinTopicData.ContentFilter *(class in rti.connexdds)*, 426
- PublicationBuiltinTopicData.ContentFilteredTopic *(class in rti.connexdds)*, 426
- PublicationBuiltinTopicData.ContentFilteredTopicSeq *(class in rti.connexdds)*, 427
- PublicationBuiltinTopicData.DataReader *(class in rti.connexdds)*, 430
- PublicationBuiltinTopicData.DataReaderListener *(class in rti.connexdds)*, 436
- PublicationBuiltinTopicData.DataReader.Selector *(class in rti.connexdds)*, 430
- PublicationBuiltinTopicData.DataReaderSeq *(class in rti.connexdds)*, 437
- PublicationBuiltinTopicData.DataWriter *(class in rti.connexdds)*, 439
- PublicationBuiltinTopicData.DataWriterListener *(class in rti.connexdds)*, 447
- PublicationBuiltinTopicData.DataWriterSeq *(class in rti.connexdds)*, 449
- PublicationBuiltinTopicData.ITopicDescription *(class in rti.connexdds)*, 451
- PublicationBuiltinTopicData.LoanedSample *(class in rti.connexdds)*, 451
- PublicationBuiltinTopicData.LoanedSamples *(class in rti.connexdds)*, 452
- PublicationBuiltinTopicData.NoOpDataReaderListener *(class in rti.connexdds)*, 452
- PublicationBuiltinTopicData.NoOpDataWriterListener *(class in rti.connexdds)*, 453
- PublicationBuiltinTopicData.NoOpTopicListener *(class in rti.connexdds)*, 455
- PublicationBuiltinTopicData.Sample *(class in rti.connexdds)*, 455
- PublicationBuiltinTopicDataSeq *(class in rti.connexdds)*, 464
- PublicationBuiltinTopicData.SharedSamples *(class in rti.connexdds)*, 456
- PublicationBuiltinTopicData.TimestampedSeq *(class in rti.connexdds)*, 466
- PublicationBuiltinTopicData.Topic *(class in rti.connexdds)*, 456
- PublicationBuiltinTopicData.TopicDescription *(class in rti.connexdds)*, 457
- PublicationBuiltinTopicData.TopicListener *(class in rti.connexdds)*, 457
- PublicationBuiltinTopicData.TopicSeq *(class in rti.connexdds)*, 458
- PublicationBuiltinTopicData.ValidLoanedSamples *(class in rti.connexdds)*, 460
- PublicationBuiltinTopicData.WriterContentFilter *(class in rti.connexdds)*, 460
- PublicationBuiltinTopicData.WriterContentFilterHelper *(class in rti.connexdds)*, 461
- PublicationMatchedStatus *(class in rti.connexdds)*, 469
- publish_mode *(rti.connexdds.DataWriterQos property)*, 131
- Publisher *(class in rti.connexdds)*, 472
- publisher *(rti.connexdds.DataWriter property)*, 118
- publisher *(rti.connexdds.DynamicData.DataWriter property)*, 201
- publisher *(rti.connexdds.IAnyDataWriter property)*, 294
- publisher *(rti.connexdds.ParticipantBuiltinTopicData.DataWriter property)*, 387
- publisher *(rti.connexdds.PublicationBuiltinTopicData.DataWriter property)*, 444
- publisher *(rti.connexdds.ServiceRequest.DataWriter property)*, 540
- publisher *(rti.connexdds.SubscriptionBuiltinTopicData.DataWriter property)*, 602
- publisher *(rti.connexdds.TopicBuiltinTopicData.DataWriter*

- property*), 657
 - `publisher_group_data_max_length` (*rti.connexdds.DomainParticipantResourceLimits property*), 174
 - `publisher_key` (*rti.connexdds.PublicationBuiltinTopicData property*), 463
 - `publisher_qos` (*rti.connexdds.QosProvider property*), 483
 - `publisher_qos_from_profile()` (*rti.connexdds.QosProvider method*), 483
 - `publisher_qos_profile_name` (*rti.connexdds.MonitoringDistributionSettings property*), 351
 - `PublisherListener` (*class in rti.connexdds*), 473
 - `PublisherQos` (*class in rti.connexdds*), 474
 - `PublisherSeq` (*class in rti.connexdds*), 476
 - `PublishMode` (*class in rti.connexdds*), 469
 - `PublishModeKind` (*class in rti.connexdds*), 470
 - `PublishModeKind.PublishModeKind` (*class in rti.connexdds*), 470
 - `pulled_fragment_bytes` (*rti.connexdds.DataWriterProtocolStatus property*), 123
 - `pulled_fragment_count` (*rti.connexdds.DataWriterProtocolStatus property*), 123
 - `pulled_sample_bytes` (*rti.connexdds.DataWriterProtocolStatus property*), 123
 - `pulled_sample_count` (*rti.connexdds.DataWriterProtocolStatus property*), 123
 - `push_on_write` (*rti.connexdds.DataWriterProtocol property*), 122
 - `pushed_fragment_bytes` (*rti.connexdds.DataWriterProtocolStatus property*), 123
 - `pushed_fragment_count` (*rti.connexdds.DataWriterProtocolStatus property*), 123
 - `pushed_sample_bytes` (*rti.connexdds.DataWriterProtocolStatus property*), 123
 - `pushed_sample_count` (*rti.connexdds.DataWriterProtocolStatus property*), 124
- Q**
- `qos` (*rti.connexdds.DataReader property*), 88
 - `qos` (*rti.connexdds.DataWriter property*), 118
 - `qos` (*rti.connexdds.DomainParticipant property*), 162
 - `qos` (*rti.connexdds.DynamicData.DataReader property*), 192
 - `qos` (*rti.connexdds.DynamicData.DataWriter property*), 201
 - `qos` (*rti.connexdds.DynamicData.Topic property*), 215
 - `qos` (*rti.connexdds.IAnyDataReader property*), 291
 - `qos` (*rti.connexdds.IAnyDataWriter property*), 294
 - `qos` (*rti.connexdds.IAnyTopic property*), 297
 - `qos` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader property*), 378
 - `qos` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter property*), 387
 - `qos` (*rti.connexdds.ParticipantBuiltinTopicData.Topic property*), 400
 - `qos` (*rti.connexdds.PublicationBuiltinTopicData.DataReader property*), 434
 - `qos` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter property*), 444
 - `qos` (*rti.connexdds.PublicationBuiltinTopicData.Topic property*), 457
 - `qos` (*rti.connexdds.Publisher property*), 473
 - `qos` (*rti.connexdds.ServiceRequest.DataReader property*), 532
 - `qos` (*rti.connexdds.ServiceRequest.DataWriter property*), 540
 - `qos` (*rti.connexdds.ServiceRequest.Topic property*), 553
 - `qos` (*rti.connexdds.Subscriber property*), 580
 - `qos` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader property*), 593
 - `qos` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter property*), 602
 - `qos` (*rti.connexdds.SubscriptionBuiltinTopicData.Topic property*), 615
 - `qos` (*rti.connexdds.Topic property*), 639
 - `qos` (*rti.connexdds.TopicBuiltinTopicData.DataReader property*), 648
 - `qos` (*rti.connexdds.TopicBuiltinTopicData.DataWriter property*), 657
 - `qos` (*rti.connexdds.TopicBuiltinTopicData.Topic property*), 670
 - `QOS_ELEMENT_NAME_USE_XML_CONFIG` (*rti.connexdds.DomainParticipantConfigParams attribute*), 163
 - `qos_library` (*rti.logging.distlog.LoggerOptions property*), 772
 - `qos_profile` (*rti.logging.distlog.LoggerOptions property*), 772
 - `qos_profile_libraries` (*rti.connexdds.QosProvider property*), 483
 - `qos_profiles()` (*rti.connexdds.QosProvider method*), 483
 - `QosPolicyCount` (*class in rti.connexdds*), 478
 - `QosPolicyCountSeq` (*class in rti.connexdds*), 478
 - `QosPrintFormat` (*class in rti.connexdds*), 480
 - `QosProvider` (*class in rti.connexdds*), 481
 - `QosProviderParams` (*class in rti.connexdds*), 484
 - `Query` (*class in rti.connexdds*), 485
 - `query_condition_allocation` (*rti.connexdds.DomainParticipantResourceLimits property*), 174
 - `QueryCondition` (*class in rti.connexdds*), 485
 - `queue_size` (*rti.logging.distlog.LoggerOptions property*), 772
 - `QUEUEING` (*rti.connexdds.ServiceKind attribute*), 521
 - `QUEUEING` (*rti.connexdds.ServiceKind.ServiceKind attribute*), 522
 - `quorum_count` (*rti.connexdds.EndpointGroup property*), 256
- R**
- `Rank` (*class in rti.connexdds*), 486
 - `rank` (*rti.connexdds.SampleInfo property*), 509
 - `reachability_lease_duration` (*rti.connexdds.ParticipantBuiltinTopicData property*), 406
 - `READ` (*rti.connexdds.SampleState attribute*), 516

`read()` (*rti.connexdds.DataReader* method), 88
`read()` (*rti.connexdds.DataReader.Selector* method), 84
`read()` (*rti.connexdds.DynamicData.DataReader* method), 193
`read()` (*rti.connexdds.DynamicData.DataReader.Selector* method), 189
`read()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* method), 378
`read()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector* method), 374
`read()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader* method), 434
`read()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector* method), 430
`read()` (*rti.connexdds.ServiceRequest.DataReader* method), 532
`read()` (*rti.connexdds.ServiceRequest.DataReader.Selector* method), 528
`read()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader* method), 593
`read()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector* method), 589
`read()` (*rti.connexdds.TopicBuiltinTopicData.DataReader* method), 648
`read()` (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector* method), 644
`read_loaned()` (*rti.connexdds.DataReader* method), 89
`read_loaned()` (*rti.connexdds.DataReader.Selector* method), 85
`read_loaned()` (*rti.connexdds.DynamicData.DataReader* method), 193
`read_loaned()` (*rti.connexdds.DynamicData.DataReader.Selector* method), 189
`read_loaned()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* method), 378
`read_loaned()` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector* method), 374
`read_loaned()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader* method), 435
`read_loaned()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector* method), 430
`read_loaned()` (*rti.connexdds.ServiceRequest.DataReader* method), 532
`read_loaned()` (*rti.connexdds.ServiceRequest.DataReader.Selector* method), 528
`read_loaned()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader* method), 593
`read_loaned()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector* method), 589
`read_loaned()` (*rti.connexdds.TopicBuiltinTopicData.DataReader* method), 648
`read_loaned()` (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector* method), 644
`read_replies()` (*rti.rpc.Requester* method), 762
`read_requests()` (*rti.rpc.Replier* method), 765
`ReadCondition` (class in *rti.connexdds*), 487
`reader_checkpoint_frequency` (*rti.connexdds.PersistentStorageSettings* property), 413
`reader_data_lifecycle` (*rti.connexdds.DataReaderQos* property), 103
`reader_data_tag_list_max_length` (*rti.connexdds.DomainParticipantResourceLimits*

property), 174
 reader_data_tag_string_max_length (*rti.connexdds.DomainParticipantResourceLimits property*), 174
 reader_property_list_max_length (*rti.connexdds.DomainParticipantResourceLimits property*), 174
 reader_property_string_max_length (*rti.connexdds.DomainParticipantResourceLimits property*), 174
 reader_resource_limits (*rti.connexdds.DataReaderQos property*), 103
 reader_user_data_max_length (*rti.connexdds.DomainParticipantResourceLimits property*), 174
 ReaderDataLifecycle (*class in rti.connexdds*), 487
 REALTIME_PRIORITY (*rti.connexdds.ThreadSettingsKindMask attribute*), 633
 reassembled_sample_count (*rti.connexdds.DataReaderProtocolStatus property*), 96
 receive_address (*rti.connexdds.TransportMulticastSettings property*), 708
 receive_port (*rti.connexdds.TransportMulticastSettings property*), 708
 receive_port (*rti.connexdds.TransportUnicastSettings property*), 713
 receive_replies() (*rti.rpc.Requester method*), 762
 receive_requests() (*rti.rpc.Replier method*), 765
 receive_window_size (*rti.connexdds.RtpsReliableReaderProtocol property*), 498
 received_ack_bytes (*rti.connexdds.DataWriterProtocolStatus property*), 124
 received_ack_count (*rti.connexdds.DataWriterProtocolStatus property*), 124
 received_fragment_count (*rti.connexdds.DataReaderProtocolStatus property*), 96
 received_gap_bytes (*rti.connexdds.DataReaderProtocolStatus property*), 96
 received_gap_count (*rti.connexdds.DataReaderProtocolStatus property*), 96
 received_heartbeat_bytes (*rti.connexdds.DataReaderProtocolStatus property*), 96
 received_heartbeat_count (*rti.connexdds.DataReaderProtocolStatus property*), 96
 received_nack_bytes (*rti.connexdds.DataWriterProtocolStatus property*), 124
 received_nack_count (*rti.connexdds.DataWriterProtocolStatus property*), 124
 received_nack_fragment_bytes (*rti.connexdds.DataWriterProtocolStatus property*), 124
 received_nack_fragment_count (*rti.connexdds.DataWriterProtocolStatus property*), 124
 received_sample_bytes (*rti.connexdds.DataReaderProtocolStatus property*), 96
 received_sample_count (*rti.connexdds.DataReaderProtocolStatus property*), 96
 receiver_pool (*rti.connexdds.DomainParticipantQos property*), 170
 ReceiverPool (*class in rti.connexdds*), 489
 reception_sequence_number (*rti.connexdds.SampleInfo property*), 509
 reception_timestamp (*rti.connexdds.SampleInfo property*), 509
 RECORDING (*rti.connexdds.ServiceKind attribute*), 521
 RECORDING (*rti.connexdds.ServiceKind.ServiceKind attribute*), 522
 RECOVER_STATE (*rti.connexdds.InstanceStateConsistencyKind attribute*), 314
 REDELIVERED (*rti.connexdds.SampleFlag attribute*), 505
 register_content_filter() (*rti.connexdds.DomainParticipant method*), 162
 register_durable_subscription() (*rti.connexdds.DomainParticipant method*), 162
 register_idl_type() (*rti.connexdds.DomainParticipant static method*), 162
 register_instance() (*rti.connexdds.DataWriter method*), 118
 register_instance() (*rti.connexdds.DynamicData.DataWriter method*), 201
 register_instance() (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*), 387
 register_instance() (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*), 444
 register_instance() (*rti.connexdds.ServiceRequest.DataWriter method*), 540
 register_instance() (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method*), 602
 register_instance() (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 657
 register_type() (*rti.connexdds.DomainParticipant method*), 162
 register_type() (*rti.connexdds.DynamicData.TopicTypeSupport static method*), 218
 REJECTED_BY_DECODE_FAILURE (*rti.connexdds.SampleRejectedState attribute*), 513

- REJECTED_BY_INSTANCES_LIMIT
(*rti.connexdds.SampleRejectedState* attribute), 513
- REJECTED_BY_REMOTE_WRITERS_PER_VIRTUAL_QUEUE_LIMIT
(*rti.connexdds.SampleRejectedState* attribute), 513
- REJECTED_BY_SAMPLES_LIMIT
(*rti.connexdds.SampleRejectedState* attribute), 513
- REJECTED_BY_SAMPLES_PER_INSTANCE_LIMIT
(*rti.connexdds.SampleRejectedState* attribute), 513
- REJECTED_BY_SAMPLES_PER_REMOTE_WRITER_LIMIT
(*rti.connexdds.SampleRejectedState* attribute), 513
- rejected_sample_count
(*rti.connexdds.DataReaderProtocolStatus* property), 96
- rejected_sample_count
(*rti.connexdds.DataWriterProtocolStatus* property), 124
- related_original_publication_virtual_guid
(*rti.connexdds.SampleInfo* property), 509
- related_original_publication_virtual_sample_identity (*rti.connexdds.SampleInfo* property), 509
- related_original_publication_virtual_sequence_number (*rti.connexdds.SampleInfo* property), 509
- related_reader_guid (*rti.connexdds.WriteParams* property), 755
- related_sample_identity
(*rti.connexdds.FilterSampleInfo* property), 272
- related_sample_identity (*rti.connexdds.WriteParams* property), 755
- related_source_guid (*rti.connexdds.SampleInfo* property), 509
- related_source_guid (*rti.connexdds.WriteParams* property), 755
- related_subscription_guid
(*rti.connexdds.SampleInfo* property), 509
- related_topic_name (*rti.connexdds.ContentFilterProperty* property), 76
- related_type() (*rti.connexdds.AliasType* method), 41
- release_version (*rti.connexdds.ProductVersion* property), 423
- Reliability (class in *rti.connexdds*), 489
- reliability (*rti.connexdds.DataReaderQos* property), 103
- reliability (*rti.connexdds.DataWriterQos* property), 132
- reliability (*rti.connexdds.PublicationBuiltinTopicData* property), 463
- reliability (*rti.connexdds.SubscriptionBuiltinTopicData* property), 621
- reliability (*rti.connexdds.TopicBuiltinTopicData* property), 675
- reliability (*rti.connexdds.TopicQos* property), 685
- ReliabilityKind (class in *rti.connexdds*), 490
- ReliabilityKind.ReliabilityKind (class in *rti.connexdds*), 490
- RELIABLE (*rti.connexdds.ReliabilityKind* attribute), 490
- RELIABLE (*rti.connexdds.ReliabilityKind.ReliabilityKind* attribute), 491
- reliable() (*rti.connexdds.Reliability* static method), 490
- RELIABLE_READER_ACTIVITY_CHANGED
(*rti.connexdds.StatusMask* attribute), 567
- reliable_reader_activity_changed_status
(*rti.connexdds.DataWriter* property), 118
- reliable_reader_activity_changed_status
(*rti.connexdds.DynamicData.DataWriter* property), 202
- reliable_reader_activity_changed_status
(*rti.connexdds.ParticipantBuiltinTopicData.DataWriter* property), 388
- reliable_reader_activity_changed_status
(*rti.connexdds.PublicationBuiltinTopicData.DataWriter* property), 444
- reliable_reader_activity_changed_status
(*rti.connexdds.ServiceRequest.DataWriter* property), 541
- reliable_reader_activity_changed_status
(*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* property), 602
- reliable_reader_activity_changed_status
(*rti.connexdds.TopicBuiltinTopicData.DataWriter* property), 657
- RELIABLE_WRITER_CACHE_CHANGED
(*rti.connexdds.StatusMask* attribute), 567
- reliable_writer_cache_changed_status
(*rti.connexdds.DataWriter* property), 118
- reliable_writer_cache_changed_status
(*rti.connexdds.DynamicData.DataWriter* property), 202
- reliable_writer_cache_changed_status (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter* property), 388
- reliable_writer_cache_changed_status (*rti.connexdds.PublicationBuiltinTopicData.DataWriter* property), 444
- reliable_writer_cache_changed_status
(*rti.connexdds.ServiceRequest.DataWriter* property), 541
- reliable_writer_cache_changed_status (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* property), 602
- reliable_writer_cache_changed_status
(*rti.connexdds.TopicBuiltinTopicData.DataWriter* property), 657
- ReliableReaderActivityChangedStatus (class in *rti.connexdds*), 492
- ReliableWriterCacheChangedStatus (class in *rti.connexdds*), 492
- reload_profiles() (*rti.connexdds.QosProvider* method), 483
- remote_administration_enabled
(*rti.logging.distlog.LoggerOptions* property), 773
- remote_participant_allocation
(*rti.connexdds.DomainParticipantResourceLimits* property), 174

remote_participant_hash_buckets
(*rti.connexdds.DomainParticipantResourceLimits*
property), 174

remote_participant_purge_kind
(*rti.connexdds.DiscoveryConfig* property), 150

remote_reader_allocation
(*rti.connexdds.DomainParticipantResourceLimits*
property), 174

remote_reader_hash_buckets
(*rti.connexdds.DomainParticipantResourceLimits*
property), 174

remote_topic_query_allocation
(*rti.connexdds.DomainParticipantResourceLimits*
property), 174

remote_topic_query_hash_buckets
(*rti.connexdds.DomainParticipantResourceLimits*
property), 174

remote_writer_allocation
(*rti.connexdds.DomainParticipantResourceLimits*
property), 175

remote_writer_hash_buckets
(*rti.connexdds.DomainParticipantResourceLimits*
property), 175

RemoteParticipantPurgeKind (class in *rti.connexdds*),
493

RemoteParticipantPurgeKind.RemotePartici-
pantPurgeKind (class in *rti.connexdds*),
493

RemoteUnknownExceptionError, 769

RemoteUnknownOperationError, 769

remove () (*rti.connexdds.AnyDataReaderSeq* method), 45

remove () (*rti.connexdds.AnyDataWriterSeq* method), 48

remove () (*rti.connexdds.AnyTopicSeq* method), 50

remove () (*rti.connexdds.BoolSeq* method), 56

remove () (*rti.connexdds.ChannelSettingsSeq* method), 65

remove () (*rti.connexdds.CharSeq* method), 68

remove () (*rti.connexdds.ConditionSeq* method), 75

remove () (*rti.connexdds.ContentFilteredTopicSeq* method), 79

remove () (*rti.connexdds.CookieSeq* method), 82

remove () (*rti.connexdds.DataReaderSeq* method), 109

remove () (*rti.connexdds.DataTag* method), 113

remove () (*rti.connexdds.DataWriterSeq* method), 139

remove () (*rti.connexdds.DomainParticipantSeq* method), 178

remove ()
(*rti.connexdds.DynamicData.ContentFilteredTopicSeq*
method), 188

remove () (*rti.connexdds.DynamicData.DataReaderSeq*
method), 197

remove () (*rti.connexdds.DynamicData.DataWriterSeq*
method), 208

remove () (*rti.connexdds.DynamicDataSeq* method), 251

remove () (*rti.connexdds.DynamicDataTimestampedSeq*
method), 254

remove () (*rti.connexdds.DynamicData.TopicSeq* method),
218

remove () (*rti.connexdds.EndpointGroupSeq* method), 258

remove () (*rti.connexdds.EntitySeq* method), 263

remove () (*rti.connexdds.EnumMemberSeq* method), 265

remove () (*rti.connexdds.Float32Seq* method), 276

remove () (*rti.connexdds.Float64Seq* method), 279

remove () (*rti.connexdds.Float128Seq* method), 274

remove () (*rti.connexdds.IAnyDataReaderSeq* method), 293

remove () (*rti.connexdds.IAnyDataWriterSeq* method), 296

remove () (*rti.connexdds.IAnyTopicSeq* method), 299

remove () (*rti.connexdds.IConditionSeq* method), 301

remove () (*rti.connexdds.IEntitySeq* method), 304

remove () (*rti.connexdds.InstanceHandleSeq* method), 311

remove () (*rti.connexdds.Int8Seq* method), 325

remove () (*rti.connexdds.Int16Seq* method), 316

remove () (*rti.connexdds.Int32Seq* method), 319

remove () (*rti.connexdds.Int64Seq* method), 323

remove () (*rti.connexdds.LocatorFilterElementSeq* method),
335

remove () (*rti.connexdds.LocatorSeq* method), 340

remove () (*rti.connexdds.MemberSeq* method), 349

remove () (*rti.connexdds.MonitoringMetricSelectionSeq*
method), 356

remove () (*rti.connexdds.MulticastMappingSeq* method), 360

remove () (*rti.connexdds.ParticipantBuiltinTopicData.Content-*
FilteredTopicSeq method),
373

remove () (*rti.connexdds.ParticipantBuiltinTopic-*
Data.DataReaderSeq method),
383

remove () (*rti.connexdds.ParticipantBuiltinTopic-*
Data.DataWriterSeq method),
394

remove () (*rti.connexdds.ParticipantBuiltinTopicDataSeq*
method), 408

remove () (*rti.connexdds.ParticipantBuiltinTopicDataTimes-*
tampedSeq method),
411

remove () (*rti.connexdds.ParticipantBuiltinTopicData.TopicSeq*
method), 403

remove () (*rti.connexdds.Property* method), 424

remove () (*rti.connexdds.PublicationBuiltinTopicData.Content-*
FilteredTopicSeq method),
429

remove () (*rti.connexdds.PublicationBuiltinTopic-*
Data.DataReaderSeq method),
439

remove () (*rti.connexdds.PublicationBuiltinTopic-*
Data.DataWriterSeq method),
451

remove () (*rti.connexdds.PublicationBuiltinTopicDataSeq*
method), 466

remove () (*rti.connexdds.PublicationBuiltinTopicDataTimes-*
tampedSeq method),
469

remove ()
(*rti.connexdds.PublicationBuiltinTopicData.TopicSeq*
method), 460

remove () (*rti.connexdds.PublisherSeq* method), 478

remove () (*rti.connexdds.QosPolicyCountSeq* method), 480

remove () (*rti.connexdds.ServiceRequest.ContentFilteredTopic-*
Seq method),
527

- remove () (*rti.connexdds.ServiceRequest.DataReaderSeq method*), 536
- remove () (*rti.connexdds.ServiceRequest.DataWriterSeq method*), 547
- remove () (*rti.connexdds.ServiceRequestSeq method*), 562
- remove () (*rti.connexdds.ServiceRequestTimestampedSeq method*), 565
- remove () (*rti.connexdds.ServiceRequest.TopicSeq method*), 555
- remove () (*rti.connexdds.StringPairSeq method*), 574
- remove () (*rti.connexdds.StringSeq method*), 576
- remove () (*rti.connexdds.SubscriberSeq method*), 584
- remove () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopicSeq method*), 588
- remove () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReaderSeq method*), 598
- remove () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriterSeq method*), 609
- remove () (*rti.connexdds.SubscriptionBuiltinTopicDataSeq method*), 625
- remove () (*rti.connexdds.SubscriptionBuiltinTopicDataTimestampedSeq method*), 627
- remove () (*rti.connexdds.SubscriptionBuiltinTopicData.TopicSeq method*), 618
- remove () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopicSeq method*), 643
- remove () (*rti.connexdds.TopicBuiltinTopicData.DataReaderSeq method*), 653
- remove () (*rti.connexdds.TopicBuiltinTopicData.DataWriterSeq method*), 664
- remove () (*rti.connexdds.TopicBuiltinTopicDataSeq method*), 678
- remove () (*rti.connexdds.TopicBuiltinTopicDataTimestampedSeq method*), 680
- remove () (*rti.connexdds.TopicBuiltinTopicData.TopicSeq method*), 673
- remove () (*rti.connexdds.TopicSeq method*), 692
- remove () (*rti.connexdds.TransportInfoSeq method*), 701
- remove () (*rti.connexdds.TransportMulticastSeq method*), 708
- remove () (*rti.connexdds.TransportMulticastSettingsSeq method*), 711
- remove () (*rti.connexdds.TransportUnicastSettingsSeq method*), 716
- remove () (*rti.connexdds.Uint8Seq method*), 736
- remove () (*rti.connexdds.Uint16Seq method*), 728
- remove () (*rti.connexdds.Uint32Seq method*), 731
- remove () (*rti.connexdds.Uint64Seq method*), 733
- remove () (*rti.connexdds.UnionMemberSeq method*), 740
- remove () (*rti.connexdds.WcharSeq method*), 751
- remove () (*rti.connexdds.WstringSeq method*), 758
- remove_from_expression_parameter () (*rti.connexdds.ContentFilteredTopic method*), 77
- remove_from_expression_parameter () (*rti.connexdds.DynamicData.ContentFilteredTopic method*), 185
- remove_from_expression_parameter () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic method*), 371
- remove_from_expression_parameter () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic method*), 427
- remove_from_expression_parameter () (*rti.connexdds.ServiceRequest.ContentFilteredTopic method*), 525
- remove_from_expression_parameter () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic method*), 585
- remove_from_expression_parameter () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic method*), 641
- remove_peer () (*rti.connexdds.DomainParticipant method*), 163
- remove_peers () (*rti.connexdds.DomainParticipant method*), 163
- replace_automatic_values (*rti.connexdds.WriteParams property*), 755
- replace_empty_instances (*rti.connexdds.DataWriterResourceLimits property*), 134
- replaced_dropped_sample_count (*rti.connexdds.DataReaderCacheStatus property*), 91
- replaced_unacknowledged_sample_count (*rti.connexdds.ReliableWriterCacheChangedStatus property*), 493
- REPLAY (*rti.connexdds.ServiceKind attribute*), 521
- REPLAY (*rti.connexdds.ServiceKind.ServiceKind attribute*), 522
- REPLICATE (*rti.connexdds.SampleFlag attribute*), 505
- Replier (*class in rti.rpc*), 764
- reply_datareader (*rti.rpc.Requester property*), 762
- reply_datawriter (*rti.rpc.Replier property*), 766
- representation (*rti.connexdds.PublicationBuiltinTopicData property*), 463
- representation (*rti.connexdds.SubscriptionBuiltinTopicData property*), 621
- representation (*rti.connexdds.TopicBuiltinTopicData property*), 675
- request_body (*rti.connexdds.ServiceRequest property*), 557
- request_datareader (*rti.rpc.Replier property*), 766
- request_datawriter (*rti.rpc.Requester property*), 763
- REQUESTED_DEADLINE_MISSED (*rti.connexdds.StatusMask attribute*), 567
- requested_deadline_missed_status (*rti.connexdds.DataReader property*), 89
- requested_deadline_missed_status

(rti.connexdds.DynamicData.DataReader property),
 193
 requested_deadline_missed_status (*rti.connexdds.ParticipantBuiltinTopicData.DataReader property*), 378
 requested_deadline_missed_status (*rti.connexdds.PublicationBuiltinTopicData.DataReader property*), 435
 requested_deadline_missed_status (*rti.connexdds.ServiceRequest.DataReader property*), 532
 requested_deadline_missed_status (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader property*), 593
 requested_deadline_missed_status (*rti.connexdds.TopicBuiltinTopicData.DataReader property*), 648
 REQUESTED_INCOMPATIBLE_QOS (*rti.connexdds.StatusMask attribute*), 567
 requested_incompatible_qos_status (*rti.connexdds.DataReader property*), 89
 requested_incompatible_qos_status (*rti.connexdds.DynamicData.DataReader property*), 193
 requested_incompatible_qos_status (*rti.connexdds.ParticipantBuiltinTopicData.DataReader property*), 378
 requested_incompatible_qos_status (*rti.connexdds.PublicationBuiltinTopicData.DataReader property*), 435
 requested_incompatible_qos_status (*rti.connexdds.ServiceRequest.DataReader property*), 532
 requested_incompatible_qos_status (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader property*), 593
 requested_incompatible_qos_status (*rti.connexdds.TopicBuiltinTopicData.DataReader property*), 648
 RequestedDeadlineMissedStatus (*class in rti.connexdds*), 495
 RequestedIncompatibleQosStatus (*class in rti.connexdds*), 496
 Requester (*class in rti.rpc*), 760
 required_matched_endpoint_groups (*rti.connexdds.Availability property*), 53
 RESERVED (*rti.connexdds.LocatorKind attribute*), 337
 RESERVED (*rti.connexdds.LocatorKind.LocatorKind attribute*), 336
 RESERVED_RANGE (*rti.connexdds.TransportClassId attribute*), 696
 RESERVED_RANGE (*rti.connexdds.TransportClassId.TransportClassId attribute*), 697
 reset () (*rti.connexdds.ActivityContextMask method*), 40
 reset () (*rti.connexdds.CompressionIdMask method*), 72
 reset () (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method*), 153
 reset () (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method*), 156
 reset () (*rti.connexdds.InstanceState method*), 313
 reset () (*rti.connexdds.RtpsReservedPortKindMask method*), 503
 reset () (*rti.connexdds.SampleFlag method*), 506
 reset () (*rti.connexdds.SampleLostState method*), 512
 reset () (*rti.connexdds.SampleRejectedState method*), 515
 reset () (*rti.connexdds.SampleState method*), 518
 reset () (*rti.connexdds.StatusMask method*), 569
 reset () (*rti.connexdds.StreamKind method*), 571
 reset () (*rti.connexdds.ThreadSettingsKindMask method*), 635
 reset () (*rti.connexdds.TransportBuiltinMask method*), 695
 reset () (*rti.connexdds.ViewState method*), 746
 reset () (*rti.connexdds.WriteParams method*), 756
 reset_default () (*rti.connexdds.QosProvider static method*), 483
 reset_handler () (*rti.connexdds.GuardCondition method*), 286
 reset_handler () (*rti.connexdds.QueryCondition method*), 486
 reset_handler () (*rti.connexdds.ReadCondition method*), 487
 reset_handler () (*rti.connexdds.StatusCondition method*), 566
 reset_output_handler () (*rti.connexdds.Logger method*), 345
 resize () (*rti.connexdds.ByteVector method*), 61
 resize () (*rti.connexdds.CharSeq method*), 68
 resize () (*rti.connexdds.CookieVector method*), 83
 resize () (*rti.connexdds.EndpointGroupVector method*), 259
 resize () (*rti.connexdds.Float32Seq method*), 276
 resize () (*rti.connexdds.Float64Seq method*), 279
 resize () (*rti.connexdds.Int8Seq method*), 325
 resize () (*rti.connexdds.Int16Seq method*), 316
 resize () (*rti.connexdds.Int32Seq method*), 319
 resize () (*rti.connexdds.Int32Vector method*), 320
 resize () (*rti.connexdds.Int64Seq method*), 323
 resize () (*rti.connexdds.LocatorVector method*), 341
 resize () (*rti.connexdds.TransportInfoVector method*), 702
 resize () (*rti.connexdds.Uint8Seq method*), 736
 resize () (*rti.connexdds.Uint16Seq method*), 728
 resize () (*rti.connexdds.Uint32Seq method*), 731
 resize () (*rti.connexdds.Uint64Seq method*), 734
 resolve () (*rti.connexdds.AliasType method*), 41
 resolve_type () (*rti.connexdds.AliasType static method*), 41
 resource_limits (*rti.connexdds.DataReaderQos property*), 103
 resource_limits (*rti.connexdds.DataWriterQos property*), 132
 resource_limits (*rti.connexdds.DomainParticipantQos property*), 170
 resource_limits (*rti.connexdds.TopicBuiltinTopicData property*), 675
 resource_limits (*rti.connexdds.TopicQos property*), 685
 resource_selection (*rti.connexdds.MonitoringMetricSelection property*),

- 353
- ResourceLimits (class in *rti.connexdds*), 496
- response_data (*rti.connexdds.AcknowledgmentInfo* property), 35
- restore (*rti.connexdds.PersistentStorageSettings* property), 413
- resume() (*rti.connexdds.SuspendedPublication* method), 628
- resume_endpoint_discovery() (*rti.connexdds.DomainParticipant* method), 163
- retain() (*rti.connexdds.FlowController* method), 280
- retain() (*rti.connexdds.IAnyDataReader* method), 291
- retain() (*rti.connexdds.IAnyDataWriter* method), 294
- retain() (*rti.connexdds.IEntity* method), 302
- retain() (*rti.connexdds.TopicQuery* method), 686
- return_loan() (*rti.connexdds.DataReader.LoanedSamples* method), 84
- return_loan() (*rti.connexdds.DynamicData.LoanedSamples* method), 211
- return_loan() (*rti.connexdds.LoanedDynamicData* method), 331
- return_loan() (*rti.connexdds.ParticipantBuiltinTopicData.LoanedSamples* method), 396
- return_loan() (*rti.connexdds.PublicationBuiltinTopicData.LoanedSamples* method), 452
- return_loan() (*rti.connexdds.ServiceRequest.LoanedSamples* method), 548
- return_loan() (*rti.connexdds.SubscriptionBuiltinTopicData.LoanedSamples* method), 611
- return_loan() (*rti.connexdds.TopicBuiltinTopicData.LoanedSamples* method), 665
- revision_version (*rti.connexdds.ProductVersion* property), 423
- role_name (*rti.connexdds.EndpointGroup* property), 256
- role_name (*rti.connexdds.EntityName* property), 261
- ROUND_ROBIN (*rti.connexdds.FlowControllerSchedulingPolicy* attribute), 283
- ROUND_ROBIN (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy* attribute), 282
- ROUND_ROBIN (*rti.connexdds.ThreadSettingsCpuRotationKind* attribute), 631
- ROUND_ROBIN (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* attribute), 631
- round_trip_time (*rti.connexdds.RtpsReliableReaderProtocol* property), 498
- ROUTING (*rti.connexdds.ServiceKind* attribute), 521
- ROUTING (*rti.connexdds.ServiceKind.ServiceKind* attribute), 522
- rti.asyncio module, 770
- rti.connexdds module, 31
- rti.logging module, 773
- rti.logging.distlog module, 770
- rti.logging.handler module, 773
- rti.types.builtin module, 759
- rtps_app_id (*rti.connexdds.WireProtocol* property), 752
- RTPS_AUTO_ID (*rti.connexdds.WireProtocol* attribute), 752
- RTPS_AUTO_ID_FROM_IP (*rti.connexdds.WireProtocolAutoKind* attribute), 753
- RTPS_AUTO_ID_FROM_IP (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* attribute), 753
- RTPS_AUTO_ID_FROM_MAC (*rti.connexdds.WireProtocolAutoKind* attribute), 753
- RTPS_AUTO_ID_FROM_MAC (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* attribute), 753
- RTPS_AUTO_ID_FROM_UUID (*rti.connexdds.WireProtocolAutoKind* attribute), 753
- RTPS_AUTO_ID_FROM_UUID (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* attribute), 753
- rtps_auto_id_kind (*rti.connexdds.WireProtocol* property), 752
- rtps_host_id (*rti.connexdds.WireProtocol* property), 752
- rtps_instance_id (*rti.connexdds.WireProtocol* property), 752
- rtps_object_id (*rti.connexdds.DataReaderProtocol* property), 95
- rtps_object_id (*rti.connexdds.DataWriterProtocol* property), 122
- rtps_protocol_version (*rti.connexdds.ParticipantBuiltinTopicData* property), 406
- rtps_protocol_version (*rti.connexdds.PublicationBuiltinTopicData* property), 463
- rtps_protocol_version (*rti.connexdds.SubscriptionBuiltinTopicData* property), 621
- rtps_reliable_reader (*rti.connexdds.DataReaderProtocol* property), 95
- rtps_reliable_writer (*rti.connexdds.DataWriterProtocol* property), 122
- rtps_reserved_port_mask (*rti.connexdds.WireProtocol* property), 752
- rtps_vendor_id (*rti.connexdds.ParticipantBuiltinTopicData* property), 406
- rtps_vendor_id (*rti.connexdds.PublicationBuiltinTopicData* property), 463
- rtps_vendor_id (*rti.connexdds.SubscriptionBuiltinTopicData* property), 621
- rtps_well_known_ports (*rti.connexdds.WireProtocol* property), 752

- RtpsReliableReaderProtocol (class in *rti.connexdds*), 497
- RtpsReliableWriterProtocol (class in *rti.connexdds*), 498
- RtpsReservedPortKindMask (class in *rti.connexdds*), 500
- RtpsWellKnownPorts (class in *rti.connexdds*), 504
- run () (in module *rti.asyncio*), 770
- run () (*rti.rpc.Service* method), 768
- ## S
- sample (*rti.connexdds.Rank* property), 487
- sample_count (*rti.connexdds.DataReaderCacheStatus* property), 91
- sample_count (*rti.connexdds.DataWriterCacheStatus* property), 120
- sample_count_peak
(*rti.connexdds.DataReaderCacheStatus* property), 91
- sample_count_peak (*rti.connexdds.DataWriterCacheStatus* property), 120
- sample_identity (*rti.connexdds.AcknowledgmentInfo* property), 35
- SAMPLE_LOST (*rti.connexdds.StatusMask* attribute), 567
- sample_lost_status (*rti.connexdds.DataReader* property), 89
- sample_lost_status
(*rti.connexdds.DynamicData.DataReader* property), 193
- sample_lost_status (*rti.connexdds.ParticipantBuiltin-TopicData.DataReader* property), 378
- sample_lost_status (*rti.connexdds.PublicationBuiltin-TopicData.DataReader* property), 435
- sample_lost_status
(*rti.connexdds.ServiceRequest.DataReader* property), 532
- sample_lost_status (*rti.connexdds.SubscriptionBuiltin-TopicData.DataReader* property), 593
- sample_lost_status
(*rti.connexdds.TopicBuiltinTopicData.DataReader* property), 648
- SAMPLE_REJECTED (*rti.connexdds.StatusMask* attribute), 567
- sample_rejected_status (*rti.connexdds.DataReader* property), 89
- sample_rejected_status
(*rti.connexdds.DynamicData.DataReader* property), 193
- sample_rejected_status (*rti.connexdds.Participant- BuiltinTopicData.DataReader* property), 379
- sample_rejected_status (*rti.connexdds.Publication- BuiltinTopicData.DataReader* property), 435
- sample_rejected_status
(*rti.connexdds.ServiceRequest.DataReader* property), 532
- sample_rejected_status (*rti.connexdds.Subscription- BuiltinTopicData.DataReader* property), 593
- sample_rejected_status
(*rti.connexdds.TopicBuiltinTopicData.DataReader* property), 648
- SAMPLE_REMOVED (*rti.connexdds.StatusMask* attribute), 567
- sample_state (*rti.connexdds.DataState* property), 111
- sample_state (*rti.connexdds.DataStateEx* property), 112
- SampleFlag (class in *rti.connexdds*), 505
- SampleIdentity (class in *rti.connexdds*), 507
- SampleInfo (class in *rti.connexdds*), 508
- SampleLostState (class in *rti.connexdds*), 510
- SampleLostStatus (class in *rti.connexdds*), 513
- SampleRejectedState (class in *rti.connexdds*), 513
- SampleRejectedStatus (class in *rti.connexdds*), 516
- samples_per_app_ack
(*rti.connexdds.RtpsReliableReaderProtocol* property), 498
- samples_per_period (*rti.connexdds.TopicQueryDispatch* property), 688
- samples_per_virtual_heartbeat
(*rti.connexdds.RtpsReliableWriterProtocol* property), 500
- SampleState (class in *rti.connexdds*), 516
- scheduling_policy (*rti.connexdds.FlowControllerProperty* property), 281
- scope (*rti.connexdds.DestinationOrder* property), 142
- SDP (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* attribute), 154
- SDP2 (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* attribute), 154
- sec (*rti.connexdds.Duration* property), 184
- sec (*rti.connexdds.Time* property), 637
- secure_volatile_reader
(*rti.connexdds.DiscoveryConfig* property), 150
- secure_volatile_writer
(*rti.connexdds.DiscoveryConfig* property), 150
- secure_volatile_writer_publish_mode
(*rti.connexdds.DiscoveryConfig* property), 150
- security (*rti.connexdds.LogCategory* attribute), 344
- security (*rti.connexdds.LogCategory.LogCategory* attribute), 342
- security_event_forwarding_level
(*rti.connexdds.MonitoringLoggingForwardingSettings* property), 352
- SEDP (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* attribute), 154
- select () (*rti.connexdds.DataReader* method), 89
- select () (*rti.connexdds.DynamicData.DataReader* method), 193
- select () (*rti.connexdds.ParticipantBuiltinTopic- Data.DataReader* method), 379
- select () (*rti.connexdds.PublicationBuiltinTopic- Data.DataReader* method), 435
- select () (*rti.connexdds.ServiceRequest.DataReader* method), 532

select () (rti.connexdds.SubscriptionBuiltinTopicData.DataReader method), 593
select () (rti.connexdds.TopicBuiltinTopicData.DataReader method), 649
select_all () (rti.connexdds.TopicQuery static method), 686
selection (rti.connexdds.TopicQueryData property), 687
send_reply () (rti.rpc.Replier method), 766
send_request () (rti.rpc.Requester method), 763
send_window_decrease_factor (rti.connexdds.RtpsReliableWriterProtocol property), 500
send_window_increase_factor (rti.connexdds.RtpsReliableWriterProtocol property), 500
send_window_size (rti.connexdds.DataWriterProtocolStatus property), 124
send_window_update_period (rti.connexdds.RtpsReliableWriterProtocol property), 500
sent_ack_bytes (rti.connexdds.DataReaderProtocolStatus property), 97
sent_ack_count (rti.connexdds.DataReaderProtocolStatus property), 97
sent_gap_bytes (rti.connexdds.DataWriterProtocolStatus property), 124
sent_gap_count (rti.connexdds.DataWriterProtocolStatus property), 124
sent_heartbeat_bytes (rti.connexdds.DataWriterProtocolStatus property), 124
sent_heartbeat_count (rti.connexdds.DataWriterProtocolStatus property), 124
sent_nack_bytes (rti.connexdds.DataReaderProtocolStatus property), 97
sent_nack_count (rti.connexdds.DataReaderProtocolStatus property), 97
sent_nack_fragment_bytes (rti.connexdds.DataReaderProtocolStatus property), 97
sent_nack_fragment_count (rti.connexdds.DataReaderProtocolStatus property), 97
sequence_number (rti.connexdds.SampleIdentity property), 508
SEQUENCE_TYPE (rti.connexdds.TypeKind attribute), 720
SEQUENCE_TYPE (rti.connexdds.TypeKind.TypeKind attribute), 722
SequenceNumber (class in rti.connexdds), 519
SequenceType (class in rti.connexdds), 520
serialize_key_with_dispose (rti.connexdds.DataWriterProtocol property), 122
serialized_type_object_dynamic_allocation_threshold (rti.connexdds.DomainParticipantResourceLimits property), 175
Service (class in rti.connexdds), 520
Service (class in rti.rpc), 768
service (rti.connexdds.DataReaderQos property), 103
service (rti.connexdds.DataWriterQos property), 132
service (rti.connexdds.DomainParticipantQos property), 170
service (rti.connexdds.PublicationBuiltinTopicData property), 463
service (rti.connexdds.SubscriptionBuiltinTopicData property), 622
service () (in module rti.rpc), 769
service_cleanup_delay (rti.connexdds.DurabilityService property), 182
service_forwarding_level (rti.connexdds.MonitoringLoggingForwardingSettings property), 352
service_id (rti.connexdds.ServiceRequest property), 557
service_id (rti.connexdds.ServiceRequestAcceptedStatus property), 558
SERVICE_REQUEST (rti.connexdds.DiscoveryConfigBuiltinChannelKindMask attribute), 151
SERVICE_REQUEST_ACCEPTED (rti.connexdds.StatusMask attribute), 567
service_request_accepted_status (rti.connexdds.DataWriter property), 118
service_request_accepted_status (rti.connexdds.DynamicData.DataWriter property), 202
service_request_accepted_status (rti.connexdds.ParticipantBuiltinTopicData.DataWriter property), 388
service_request_accepted_status (rti.connexdds.PublicationBuiltinTopicData.DataWriter property), 444
service_request_accepted_status (rti.connexdds.ServiceRequest.DataWriter property), 541
service_request_accepted_status (rti.connexdds.SubscriptionBuiltinTopicData.DataWriter property), 603
service_request_accepted_status (rti.connexdds.TopicBuiltinTopicData.DataWriter property), 657
service_request_reader (rti.connexdds.DiscoveryConfig property), 150
service_request_reader (rti.connexdds.DomainParticipant property), 163
service_request_writer (rti.connexdds.DiscoveryConfig property), 150
service_request_writer_data_lifecycle (rti.connexdds.DiscoveryConfig property), 150
service_request_writer_publish_mode (rti.connexdds.DiscoveryConfig property), 150
ServiceKind (class in rti.connexdds), 521
ServiceKind.ServiceKind (class in rti.connexdds), 521
ServiceRequest (class in rti.connexdds), 523
ServiceRequestAcceptedStatus (class in rti.connexdds), 558
ServiceRequest.ContentFilter (class in rti.connexdds), 523

- ServiceRequest.ContentFilteredTopic (class in *rti.connexdds*), 524
- ServiceRequest.ContentFilteredTopicSeq (class in *rti.connexdds*), 525
- ServiceRequest.DataReader (class in *rti.connexdds*), 527
- ServiceRequest.DataReaderListener (class in *rti.connexdds*), 533
- ServiceRequest.DataReader.Selector (class in *rti.connexdds*), 527
- ServiceRequest.DataReaderSeq (class in *rti.connexdds*), 534
- ServiceRequest.DataWriter (class in *rti.connexdds*), 536
- ServiceRequest.DataWriterListener (class in *rti.connexdds*), 544
- ServiceRequest.DataWriterSeq (class in *rti.connexdds*), 545
- ServiceRequestId (class in *rti.connexdds*), 558
- ServiceRequestId.ServiceRequestId (class in *rti.connexdds*), 558
- ServiceRequest.ITopicDescription (class in *rti.connexdds*), 547
- ServiceRequest.LoanedSample (class in *rti.connexdds*), 547
- ServiceRequest.LoanedSamples (class in *rti.connexdds*), 548
- ServiceRequest.NoOpDataReaderListener (class in *rti.connexdds*), 548
- ServiceRequest.NoOpDataWriterListener (class in *rti.connexdds*), 549
- ServiceRequest.NoOpTopicListener (class in *rti.connexdds*), 551
- ServiceRequest.Sample (class in *rti.connexdds*), 551
- ServiceRequestSeq (class in *rti.connexdds*), 560
- ServiceRequest.SharedSamples (class in *rti.connexdds*), 551
- ServiceRequestTimestampedSeq (class in *rti.connexdds*), 562
- ServiceRequest.Topic (class in *rti.connexdds*), 552
- ServiceRequest.TopicDescription (class in *rti.connexdds*), 553
- ServiceRequest.TopicListener (class in *rti.connexdds*), 553
- ServiceRequest.TopicSeq (class in *rti.connexdds*), 553
- ServiceRequest.ValidLoanedSamples (class in *rti.connexdds*), 556
- ServiceRequest.WriterContentFilter (class in *rti.connexdds*), 556
- ServiceRequest.WriterContentFilterHelper (class in *rti.connexdds*), 557
- set () (*rti.connexdds.ActivityContextMask* method), 40
- set () (*rti.connexdds.CompressionIdMask* method), 72
- set () (*rti.connexdds.DataTag* method), 113
- set () (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 153
- set () (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 156
- set () (*rti.connexdds.InstanceState* method), 313
- set () (*rti.connexdds.Property* method), 425
- set () (*rti.connexdds.RtpsReservedPortKindMask* method), 503
- set () (*rti.connexdds.SampleFlag* method), 507
- set () (*rti.connexdds.SampleLostState* method), 512
- set () (*rti.connexdds.SampleRejectedState* method), 515
- set () (*rti.connexdds.SampleState* method), 518
- set () (*rti.connexdds.StatusMask* method), 569
- set () (*rti.connexdds.StreamKind* method), 571
- set () (*rti.connexdds.ThreadSettingsKindMask* method), 635
- set () (*rti.connexdds.TransportBuiltinMask* method), 695
- set () (*rti.connexdds.ViewState* method), 746
- set_activity_context () (in module *rti.connexdds*), 758
- set_boolean () (*rti.connexdds.DynamicData* method), 234
- set_cdr_buffer () (*rti.connexdds.DynamicData* method), 234
- set_char () (*rti.connexdds.DynamicData* method), 234
- set_char_values () (*rti.connexdds.DynamicData* method), 235
- set_complex () (*rti.connexdds.DynamicData* method), 235
- set_complex_values () (*rti.connexdds.DynamicData* method), 235
- set_double () (*rti.connexdds.DynamicData* method), 235
- set_double_values () (*rti.connexdds.DynamicData* method), 236
- set_filter () (*rti.connexdds.ContentFilteredTopic* method), 77
- set_filter () (*rti.connexdds.DynamicData.ContentFilteredTopic* method), 186
- set_filter () (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic* method), 371
- set_filter () (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic* method), 427
- set_filter () (*rti.connexdds.ServiceRequest.ContentFilteredTopic* method), 525
- set_filter () (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic* method), 586
- set_filter () (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic* method), 641
- set_float () (*rti.connexdds.DynamicData* method), 236
- set_float32 () (*rti.connexdds.DynamicData* method), 236
- set_float32_values () (*rti.connexdds.DynamicData* method), 236
- set_float64 () (*rti.connexdds.DynamicData* method), 237
- set_float64_values () (*rti.connexdds.DynamicData* method), 237
- set_float128 () (*rti.connexdds.DynamicData* method), 236
- set_float_values () (*rti.connexdds.DynamicData* method), 237
- set_handler () (*rti.connexdds.GuardCondition* method), 286
- set_handler () (*rti.connexdds.QueryCondition* method), 486
- set_handler () (*rti.connexdds.StatusCondition* method), 566

set_handler_no_args () (*rti.connexdds.GuardCondition method*), 287
 set_handler_no_args () (*rti.connexdds.QueryCondition method*), 486
 set_handler_no_args () (*rti.connexdds.ReadCondition method*), 487
 set_int () (*rti.connexdds.DynamicData method*), 237
 set_int8 () (*rti.connexdds.DynamicData method*), 238
 set_int16 () (*rti.connexdds.DynamicData method*), 237
 set_int16_values () (*rti.connexdds.DynamicData method*), 237
 set_int32 () (*rti.connexdds.DynamicData method*), 238
 set_int32_values () (*rti.connexdds.DynamicData method*), 238
 set_int64 () (*rti.connexdds.DynamicData method*), 238
 set_int64_values () (*rti.connexdds.DynamicData method*), 238
 set_int_values () (*rti.connexdds.DynamicData method*), 239
 set_listener () (*rti.connexdds.DataReader method*), 89
 set_listener () (*rti.connexdds.Data Writer method*), 118
 set_listener () (*rti.connexdds.DomainParticipant method*), 163
 set_listener () (*rti.connexdds.DynamicData.DataReader method*), 193
 set_listener () (*rti.connexdds.DynamicData.Data Writer method*), 202
 set_listener () (*rti.connexdds.DynamicData.Topic method*), 215
 set_listener () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*), 379
 set_listener () (*rti.connexdds.ParticipantBuiltinTopicData.Data Writer method*), 388
 set_listener () (*rti.connexdds.ParticipantBuiltinTopicData.Topic method*), 400
 set_listener () (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 435
 set_listener () (*rti.connexdds.PublicationBuiltinTopicData.Data Writer method*), 444
 set_listener () (*rti.connexdds.PublicationBuiltinTopicData.Topic method*), 457
 set_listener () (*rti.connexdds.Publisher method*), 473
 set_listener () (*rti.connexdds.ServiceRequest.DataReader method*), 532
 set_listener () (*rti.connexdds.ServiceRequest.Data Writer method*), 541
 set_listener () (*rti.connexdds.ServiceRequest.Topic method*), 553
 set_listener () (*rti.connexdds.Subscriber method*), 580
 set_listener () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*), 594
 set_listener () (*rti.connexdds.SubscriptionBuiltinTopicData.Data Writer method*), 603
 set_listener () (*rti.connexdds.SubscriptionBuiltinTopicData.Topic method*), 615
 set_listener () (*rti.connexdds.Topic method*), 639
 set_listener () (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 649
 set_listener () (*rti.connexdds.TopicBuiltinTopicData.Data Writer method*), 658
 set_listener () (*rti.connexdds.TopicBuiltinTopicData.Topic method*), 670
 set_long () (*rti.connexdds.DynamicData method*), 239
 set_long_values () (*rti.connexdds.DynamicData method*), 239
 set_longdouble () (*rti.connexdds.DynamicData method*), 239
 set_longlong () (*rti.connexdds.DynamicData method*), 239
 set_longlong_values () (*rti.connexdds.DynamicData method*), 240
 set_name () (*rti.connexdds.Member method*), 347
 set_octet () (*rti.connexdds.DynamicData method*), 240
 set_octet_values () (*rti.connexdds.DynamicData method*), 240
 set_parameters () (*rti.connexdds.Filter method*), 271
 set_short () (*rti.connexdds.DynamicData method*), 240
 set_short_values () (*rti.connexdds.DynamicData method*), 240
 set_string () (*rti.connexdds.DynamicData method*), 240
 set_topic_datareader_qos () (*rti.connexdds.QosProvider method*), 483
 set_topic_datawriter_qos () (*rti.connexdds.QosProvider method*), 483
 set_topic_name_qos () (*rti.connexdds.QosProvider method*), 483
 set_uint () (*rti.connexdds.DynamicData method*), 241
 set_uint8 () (*rti.connexdds.DynamicData method*), 242
 set_uint8_values () (*rti.connexdds.DynamicData method*), 242
 set_uint16 () (*rti.connexdds.DynamicData method*), 241
 set_uint16_values () (*rti.connexdds.DynamicData method*), 241
 set_uint32 () (*rti.connexdds.DynamicData method*), 241
 set_uint32_values () (*rti.connexdds.DynamicData method*), 241
 set_uint64 () (*rti.connexdds.DynamicData method*), 242
 set_uint64_values () (*rti.connexdds.DynamicData method*), 242
 set_uint_values () (*rti.connexdds.DynamicData method*), 242
 set_ulong () (*rti.connexdds.DynamicData method*), 242
 set_ulong_values () (*rti.connexdds.DynamicData method*), 243
 set_ulonglong () (*rti.connexdds.DynamicData method*), 243

- set_ulonglong_values() (rti.connexdds.DynamicData method), 243
- set_ushort() (rti.connexdds.DynamicData method), 243
- set_ushort_values() (rti.connexdds.DynamicData method), 243
- set_value() (rti.connexdds.DynamicData method), 244
- set_values() (rti.connexdds.DynamicData method), 244
- set_wchar() (rti.connexdds.DynamicData method), 244
- set_wstring() (rti.connexdds.DynamicData method), 244
- SEVERE (rti.logging.distlog.LogLevel attribute), 771
- severe() (rti.logging.distlog.Logger static method), 772
- shared (rti.connexdds.OwnershipKind attribute), 367
- SHARED (rti.connexdds.OwnershipKind attribute), 368
- SHARED (rti.connexdds.OwnershipKind.OwnershipKind attribute), 368
- shared_ea (rti.connexdds.ExclusiveArea attribute), 268
- SHMEM (rti.connexdds.LocatorKind attribute), 337
- SHMEM (rti.connexdds.LocatorKind.LocatorKind attribute), 336
- shmem (rti.connexdds.TransportBuiltin attribute), 693
- SHMEM (rti.connexdds.TransportBuiltinMask attribute), 693
- SHMEM (rti.connexdds.TransportClassId attribute), 696
- SHMEM (rti.connexdds.TransportClassId.TransportClassId attribute), 697
- SHMEM_510 (rti.connexdds.LocatorKind attribute), 337
- SHMEM_510 (rti.connexdds.LocatorKind.LocatorKind attribute), 336
- SHMEM_510 (rti.connexdds.TransportClassId attribute), 696
- SHMEM_510 (rti.connexdds.TransportClassId.TransportClassId attribute), 697
- shmem_ref_settings (rti.connexdds.DataWriterTransferMode property), 140
- shmem_ref_transfer_mode_attached_segment_allocation (rti.connexdds.DataReaderResourceLimits property), 106
- shmem_ref_transfer_mode_max_segments (rti.connexdds.DomainParticipantResourceLimits property), 175
- ShortSeq (in module rti.connexdds), 565
- ShortType (in module rti.connexdds), 565
- shutdown_cleanup_period (rti.connexdds.Database property), 141
- shutdown_timeout (rti.connexdds.Database property), 141
- SILENT (rti.connexdds.SyslogVerbosity attribute), 629
- SILENT (rti.connexdds.Verbosity attribute), 742
- SILENT (rti.connexdds.Verbosity.Verbosity attribute), 743
- SILENT (rti.logging.distlog.LogLevel attribute), 771
- SimpleReplier (class in rti.rpc), 767
- size (rti.connexdds.ActivityContextMask property), 40
- size (rti.connexdds.CompressionIdMask property), 72
- size (rti.connexdds.DiscoveryConfigBuiltinChannelKindMask property), 153
- size (rti.connexdds.DiscoveryConfigBuiltinPluginKindMask property), 156
- size (rti.connexdds.InstanceState property), 313
- size (rti.connexdds.RtpsReservedPortKindMask property), 503
- size (rti.connexdds.SampleFlag property), 507
- size (rti.connexdds.SampleLostState property), 512
- size (rti.connexdds.SampleRejectedState property), 516
- size (rti.connexdds.SampleState property), 518
- size (rti.connexdds.StatusMask property), 569
- size (rti.connexdds.StreamKind property), 571
- size (rti.connexdds.ThreadSettingsKindMask property), 636
- size (rti.connexdds.TransportBuiltinMask property), 695
- size (rti.connexdds.ViewState property), 746
- size() (rti.connexdds.DataTag method), 114
- size() (rti.connexdds.Property method), 425
- skip_deserialization (rti.connexdds.DynamicDataTypeSerializationProperty property), 254
- sleep() (in module rti.connexdds), 759
- source_guid (rti.connexdds.SampleInfo property), 509
- source_guid (rti.connexdds.WriteParams property), 756
- source_timestamp (rti.connexdds.SampleInfo property), 509
- source_timestamp (rti.connexdds.WriteParams property), 756
- source_timestamp_resolution (rti.connexdds.Batch property), 54
- source_timestamp_tolerance (rti.connexdds.DestinationOrder property), 142
- SPDP (rti.connexdds.DiscoveryConfigBuiltinPluginKindMask attribute), 154
- SPDP2 (rti.connexdds.DiscoveryConfigBuiltinPluginKindMask attribute), 154
- spin() (in module rti.connexdds), 759
- spin_per_microsecond() (in module rti.connexdds), 759
- SQL_FILTER_NAME (rti.connexdds.Filter attribute), 271
- SQL_FILTER_NAME (rti.connexdds.MultiChannel attribute), 357
- stack_size (rti.connexdds.ThreadSettings property), 631
- state (rti.connexdds.SampleInfo property), 509
- state() (rti.connexdds.DataReader.Selector method), 85
- state() (rti.connexdds.DynamicData.DataReader.Selector method), 189
- state() (rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector method), 374
- state() (rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector method), 431
- state() (rti.connexdds.ServiceRequest.DataReader.Selector method), 528
- state() (rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector method), 589
- state() (rti.connexdds.TopicBuiltinTopicData.DataReader.Selector method), 644
- state_filter (rti.connexdds.IReadCondition property), 305
- STATUS_ALL (rti.connexdds.Verbosity attribute), 742
- STATUS_ALL (rti.connexdds.Verbosity.Verbosity attribute), 743
- status_changes (rti.connexdds.IEntity property), 302
- STATUS_LOCAL (rti.connexdds.Verbosity attribute), 742
- STATUS_LOCAL (rti.connexdds.Verbosity.Verbosity attribute), 743
- STATUS_REMOTE (rti.connexdds.Verbosity attribute), 742

- STATUS_REMOTE (*rti.connexdds.Verbosity.Verbosity attribute*), 743
- StatusCondition (*class in rti.connexdds*), 565
- StatusMask (*class in rti.connexdds*), 566
- STDIO (*rti.connexdds.ThreadSettingsKindMask attribute*), 633
- storage_settings (*rti.connexdds.Durability property*), 178
- stored_size (*rti.connexdds.DynamicDataInfo property*), 248
- stream_kind (*rti.connexdds.DataStateEx property*), 112
- StreamKind (*class in rti.connexdds*), 569
- String (*class in rti.types.builtin*), 760
- string_profile (*rti.connexdds.QosProviderParams property*), 484
- STRING_TYPE (*rti.connexdds.TypeKind attribute*), 720
- STRING_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722
- StringMap (*class in rti.connexdds*), 572
- STRINGMATCH_FILTER_NAME (*rti.connexdds.Filter attribute*), 271
- STRINGMATCH_FILTER_NAME (*rti.connexdds.MultiChannel attribute*), 357
- StringPairSeq (*class in rti.connexdds*), 572
- StringSeq (*class in rti.connexdds*), 574
- StringType (*class in rti.connexdds*), 576
- StructType (*class in rti.connexdds*), 577
- STRUCTURE_FWD_DECL_TYPE (*rti.connexdds.TypeKind attribute*), 721
- STRUCTURE_FWD_DECL_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722
- STRUCTURE_TYPE (*rti.connexdds.TypeKind attribute*), 721
- STRUCTURE_TYPE (*rti.connexdds.TypeKind.TypeKind attribute*), 722
- Subscriber (*class in rti.connexdds*), 578
- subscriber (*rti.connexdds.DataReader property*), 89
- subscriber (*rti.connexdds.DynamicData.DataReader property*), 193
- subscriber (*rti.connexdds.IAnyDataReader property*), 291
- subscriber (*rti.connexdds.ParticipantBuiltinTopicData.DataReader property*), 379
- subscriber (*rti.connexdds.PublicationBuiltinTopicData.DataReader property*), 435
- subscriber (*rti.connexdds.ServiceRequest.DataReader property*), 532
- subscriber (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader property*), 594
- subscriber (*rti.connexdds.TopicBuiltinTopicData.DataReader property*), 649
- subscriber_group_data_max_length (*rti.connexdds.DomainParticipantResourceLimits property*), 175
- subscriber_key (*rti.connexdds.SubscriptionBuiltinTopicData property*), 622
- subscriber_qos (*rti.connexdds.QosProvider property*), 483
- subscriber_qos_from_profile (*rti.connexdds.QosProvider method*), 483
- SubscriberListener (*class in rti.connexdds*), 580
- SubscriberQos (*class in rti.connexdds*), 580
- SubscriberSeq (*class in rti.connexdds*), 582
- subscription_handle (*rti.connexdds.AcknowledgmentInfo property*), 35
- SUBSCRIPTION_MATCHED (*rti.connexdds.StatusMask attribute*), 567
- subscription_matched_status (*rti.connexdds.DataReader property*), 89
- subscription_matched_status (*rti.connexdds.DynamicData.DataReader property*), 193
- subscription_matched_status (*rti.connexdds.ParticipantBuiltinTopicData.DataReader property*), 379
- subscription_matched_status (*rti.connexdds.PublicationBuiltinTopicData.DataReader property*), 435
- subscription_matched_status (*rti.connexdds.ServiceRequest.DataReader property*), 532
- subscription_matched_status (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader property*), 594
- subscription_matched_status (*rti.connexdds.TopicBuiltinTopicData.DataReader property*), 649
- subscription_name (*rti.connexdds.SubscriptionBuiltinTopicData property*), 622
- subscription_reader (*rti.connexdds.DiscoveryConfig property*), 150
- subscription_reader (*rti.connexdds.DomainParticipant property*), 163
- subscription_reader_resource_limits (*rti.connexdds.DiscoveryConfig property*), 150
- subscription_writer (*rti.connexdds.DiscoveryConfig property*), 150
- subscription_writer_data_lifecycle (*rti.connexdds.DiscoveryConfig property*), 150
- subscription_writer_publish_mode (*rti.connexdds.DiscoveryConfig property*), 150
- SubscriptionBuiltinTopicData (*class in rti.connexdds*), 584
- SubscriptionBuiltinTopicData.ContentFilter (*class in rti.connexdds*), 584
- SubscriptionBuiltinTopicData.ContentFilteredTopic (*class in rti.connexdds*), 585
- SubscriptionBuiltinTopicData.ContentFilteredTopicSeq (*class in rti.connexdds*), 586
- SubscriptionBuiltinTopicData.DataReader (*class in rti.connexdds*), 588
- SubscriptionBuiltinTopicData.DataReaderListener (*class in rti.connexdds*), 594

- SubscriptionBuiltinTopicData.DataReader.Selector (class in *rti.connexdds*), 588
- SubscriptionBuiltinTopicData.DataReaderSeq (class in *rti.connexdds*), 595
- SubscriptionBuiltinTopicData.DataWriter (class in *rti.connexdds*), 598
- SubscriptionBuiltinTopicData.DataWriterListener (class in *rti.connexdds*), 606
- SubscriptionBuiltinTopicData.DataWriterSeq (class in *rti.connexdds*), 607
- SubscriptionBuiltinTopicData.ITopicDescription (class in *rti.connexdds*), 609
- SubscriptionBuiltinTopicData.LoanedSample (class in *rti.connexdds*), 610
- SubscriptionBuiltinTopicData.LoanedSamples (class in *rti.connexdds*), 610
- SubscriptionBuiltinTopicData.NoOpDataReaderListener (class in *rti.connexdds*), 611
- SubscriptionBuiltinTopicData.NoOpDataWriterListener (class in *rti.connexdds*), 612
- SubscriptionBuiltinTopicData.NoOpTopicListener (class in *rti.connexdds*), 613
- SubscriptionBuiltinTopicData.Sample (class in *rti.connexdds*), 614
- SubscriptionBuiltinTopicDataSeq (class in *rti.connexdds*), 622
- SubscriptionBuiltinTopicData.SharedSamples (class in *rti.connexdds*), 614
- SubscriptionBuiltinTopicData.TimestampedSeq (class in *rti.connexdds*), 625
- SubscriptionBuiltinTopicData.Topic (class in *rti.connexdds*), 614
- SubscriptionBuiltinTopicData.TopicDescription (class in *rti.connexdds*), 615
- SubscriptionBuiltinTopicData.TopicListener (class in *rti.connexdds*), 616
- SubscriptionBuiltinTopicData.TopicSeq (class in *rti.connexdds*), 616
- SubscriptionBuiltinTopicData.ValidLoanedSamples (class in *rti.connexdds*), 618
- SubscriptionBuiltinTopicData.WriterContentFilter (class in *rti.connexdds*), 619
- SubscriptionBuiltinTopicData.WriterContentFilterHelper (class in *rti.connexdds*), 620
- SubscriptionMatchedStatus (class in *rti.connexdds*), 627
- SuspendedPublication (class in *rti.connexdds*), 628
- synchronization_kind (*rti.connexdds.PersistentStorageSettings* property), 413
- synchronous (*rti.connexdds.PublishMode* attribute), 470
- SYNCHRONOUS (*rti.connexdds.PublishModeKind* attribute), 471
- SYNCHRONOUS (*rti.connexdds.PublishModeKind.PublishModeKind* attribute), 470
- SyslogVerbosity (class in *rti.connexdds*), 628
- system_resource_limits (*rti.connexdds.DomainParticipantFactoryQos* property), 165
- SystemResourceLimits (class in *rti.connexdds*), 629
- ## T
- take () (*rti.connexdds.DataReader* method), 89
- take () (*rti.connexdds.DataReader.Selector* method), 85
- take () (*rti.connexdds.DynamicData.DataReader* method), 193
- take () (*rti.connexdds.DynamicData.DataReader.Selector* method), 189
- take () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* method), 379
- take () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector* method), 374
- take () (*rti.connexdds.PublicationBuiltinTopicData.DataReader* method), 435
- take () (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector* method), 431
- take () (*rti.connexdds.ServiceRequest.DataReader* method), 532
- take () (*rti.connexdds.ServiceRequest.DataReader.Selector* method), 528
- take () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader* method), 594
- take () (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector* method), 589
- take () (*rti.connexdds.TopicBuiltinTopicData.DataReader* method), 649
- take () (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector* method), 644
- take_async () (*rti.connexdds.DataReader* method), 89
- take_async () (*rti.connexdds.DynamicData.DataReader* method), 193
- take_data () (*rti.connexdds.DataReader* method), 89
- take_data () (*rti.connexdds.DataReader.Selector* method), 85
- take_data () (*rti.connexdds.DynamicData.DataReader* method), 193
- take_data () (*rti.connexdds.DynamicData.DataReader.Selector* method), 189
- take_data () (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* method), 379

take_data() (*rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 374

take_data() (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 435

take_data() (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector method*), 431

take_data() (*rti.connexdds.ServiceRequest.DataReader method*), 532

take_data() (*rti.connexdds.ServiceRequest.DataReader.Selector method*), 528

take_data() (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*), 594

take_data() (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 589

take_data() (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 649

take_data() (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector method*), 644

take_data_async() (*rti.connexdds.DataReader method*), 89

take_data_async() (*rti.connexdds.DynamicData.DataReader method*), 193

take_loaned() (*rti.connexdds.DataReader method*), 89

take_loaned() (*rti.connexdds.DataReader.Selector method*), 85

take_loaned() (*rti.connexdds.DynamicData.DataReader method*), 193

take_loaned() (*rti.connexdds.DynamicData.DataReader.Selector method*), 189

take_loaned() (*rti.connexdds.ParticipantBuiltinTopicData.DataReader method*), 379

take_loaned() (*rti.connexdds.ParticipantBuiltinTopicData.DataReader.Selector method*), 375

take_loaned() (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 435

take_loaned() (*rti.connexdds.PublicationBuiltinTopicData.DataReader.Selector method*), 431

take_loaned() (*rti.connexdds.ServiceRequest.DataReader method*), 532

take_loaned() (*rti.connexdds.ServiceRequest.DataReader.Selector method*), 528

take_loaned() (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*), 594

take_loaned() (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader.Selector method*), 589

take_loaned() (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 649

take_loaned() (*rti.connexdds.TopicBuiltinTopicData.DataReader.Selector method*), 645

take_replies() (*rti.rpc.Requester method*), 763

take_requests() (*rti.rpc.Replier method*), 766

TCPV4_LAN (*rti.connexdds.LocatorKind attribute*), 337

TCPV4_LAN (*rti.connexdds.LocatorKind.LocatorKind attribute*), 336

TCPV4_LAN (*rti.connexdds.TransportClassId attribute*), 696

TCPV4_LAN (*rti.connexdds.TransportClassId.TransportClassId attribute*), 697

TCPV4_WAN (*rti.connexdds.LocatorKind attribute*), 337

TCPV4_WAN (*rti.connexdds.LocatorKind.LocatorKind attribute*), 336

TCPV4_WAN (*rti.connexdds.TransportClassId attribute*), 696

TCPV4_WAN (*rti.connexdds.TransportClassId.TransportClassId attribute*), 697

telemetry_data (*rti.connexdds.Monitoring property*), 349

test() (*rti.connexdds.ActivityContextMask method*), 40

test() (*rti.connexdds.CompressionIdMask method*), 72

test() (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method*), 153

test() (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method*), 156

test() (*rti.connexdds.InstanceState method*), 313

test() (*rti.connexdds.RtpsReservedPortKindMask method*), 503

test() (*rti.connexdds.SampleFlag method*), 507

test() (*rti.connexdds.SampleLostState method*), 512

test() (*rti.connexdds.SampleRejectedState method*), 516

test() (*rti.connexdds.SampleState method*), 518

test() (*rti.connexdds.StatusMask method*), 569

test() (*rti.connexdds.StreamKind method*), 571

test() (*rti.connexdds.ThreadSettingsKindMask method*), 636

test() (*rti.connexdds.TransportBuiltinMask method*), 695

test() (*rti.connexdds.ViewState method*), 746

test_all() (*rti.connexdds.ActivityContextMask method*), 40

test_all() (*rti.connexdds.CompressionIdMask method*), 72

test_all() (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask method*), 153

test_all() (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask method*), 156

test_all() (*rti.connexdds.InstanceState method*), 313

test_all() (*rti.connexdds.RtpsReservedPortKindMask method*), 503

test_all() (*rti.connexdds.SampleFlag method*), 507

test_all() (*rti.connexdds.SampleLostState method*), 513

test_all() (*rti.connexdds.SampleRejectedState method*), 516

test_all() (*rti.connexdds.SampleState method*), 519

test_all() (*rti.connexdds.StatusMask method*), 569

test_all() (*rti.connexdds.StreamKind* method), 571
test_all() (*rti.connexdds.ThreadSettingsKindMask* method), 636
test_all() (*rti.connexdds.TransportBuiltinMask* method), 695
test_all() (*rti.connexdds.ViewState* method), 747
test_any() (*rti.connexdds.ActivityContextMask* method), 40
test_any() (*rti.connexdds.CompressionIdMask* method), 72
test_any() (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 153
test_any() (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 156
test_any() (*rti.connexdds.InstanceState* method), 313
test_any() (*rti.connexdds.RtpsReservedPortKindMask* method), 503
test_any() (*rti.connexdds.SampleFlag* method), 507
test_any() (*rti.connexdds.SampleLostState* method), 513
test_any() (*rti.connexdds.SampleRejectedState* method), 516
test_any() (*rti.connexdds.SampleState* method), 519
test_any() (*rti.connexdds.StatusMask* method), 569
test_any() (*rti.connexdds.StreamKind* method), 572
test_any() (*rti.connexdds.ThreadSettingsKindMask* method), 636
test_any() (*rti.connexdds.TransportBuiltinMask* method), 695
test_any() (*rti.connexdds.ViewState* method), 747
test_none() (*rti.connexdds.ActivityContextMask* method), 40
test_none() (*rti.connexdds.CompressionIdMask* method), 72
test_none() (*rti.connexdds.DiscoveryConfigBuiltinChannelKindMask* method), 153
test_none() (*rti.connexdds.DiscoveryConfigBuiltinPluginKindMask* method), 157
test_none() (*rti.connexdds.InstanceState* method), 313
test_none() (*rti.connexdds.RtpsReservedPortKindMask* method), 503
test_none() (*rti.connexdds.SampleFlag* method), 507
test_none() (*rti.connexdds.SampleLostState* method), 513
test_none() (*rti.connexdds.SampleRejectedState* method), 516
test_none() (*rti.connexdds.SampleState* method), 519
test_none() (*rti.connexdds.StatusMask* method), 569
test_none() (*rti.connexdds.StreamKind* method), 572
test_none() (*rti.connexdds.ThreadSettingsKindMask* method), 636
test_none() (*rti.connexdds.TransportBuiltinMask* method), 695
test_none() (*rti.connexdds.ViewState* method), 747
thread (*rti.connexdds.AsynchronousPublisher* property), 52
thread (*rti.connexdds.Database* property), 141
thread (*rti.connexdds.Event* property), 267
thread (*rti.connexdds.MonitoringEventDistributionSettings* property), 351
thread (*rti.connexdds.MonitoringLoggingDistributionSettings* property), 352
thread (*rti.connexdds.MonitoringPeriodicDistributionSettings* property), 356
thread (*rti.connexdds.ReceiverPool* property), 489
thread_safe_write (*rti.connexdds.Batch* property), 54
thread_settings (*rti.logging.distlog.LoggerOptions* property), 773
ThreadContext (class in *rti.connexdds*), 630
ThreadSettings (class in *rti.connexdds*), 630
ThreadSettingsCpuRotationKind (class in *rti.connexdds*), 631
ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind (class in *rti.connexdds*), 631
ThreadSettingsKindMask (class in *rti.connexdds*), 633
Time (class in *rti.connexdds*), 636
time_based_filter (*rti.connexdds.DataReaderQos* property), 103
time_based_filter (*rti.connexdds.SubscriptionBuiltinTopicData* property), 622
time_based_filter_dropped_sample_count (*rti.connexdds.DataReaderCacheStatus* property), 91
TimeBasedFilter (class in *rti.connexdds*), 638
TimeoutError, 638
timestamp (*rti.logging.distlog.MessageParams* property), 773
TIMESTAMPED (*rti.connexdds.PrintFormat* attribute), 419
TIMESTAMPED (*rti.connexdds.PrintFormat.PrintFormat* attribute), 419
TLSV4_LAN (*rti.connexdds.LocatorKind* attribute), 337
TLSV4_LAN (*rti.connexdds.LocatorKind.LocatorKind* attribute), 336
TLSV4_LAN (*rti.connexdds.TransportClassId* attribute), 696
TLSV4_LAN (*rti.connexdds.TransportClassId.TransportClassId* attribute), 697
TLSV4_WAN (*rti.connexdds.LocatorKind* attribute), 337
TLSV4_WAN (*rti.connexdds.LocatorKind.LocatorKind* attribute), 336
TLSV4_WAN (*rti.connexdds.TransportClassId* attribute), 696
TLSV4_WAN (*rti.connexdds.TransportClassId.TransportClassId* attribute), 697
to_cdr_buffer () (*rti.connexdds.DynamicData* method), 244
to_cdr_buffer () (*rti.connexdds.DynamicData.TopicTypeSupport* static method), 219
to_json () (*rti.connexdds.DynamicData* method), 244
to_microseconds () (*rti.connexdds.Duration* method), 184
to_microseconds () (*rti.connexdds.Time* method), 637
to_milliseconds () (*rti.connexdds.Duration* method), 184
to_milliseconds () (*rti.connexdds.Time* method), 638
to_seconds () (*rti.connexdds.Duration* method), 184
to_seconds () (*rti.connexdds.Time* method), 638
to_string () (*rti.connexdds.DataReaderQos* method), 103
to_string () (*rti.connexdds.DataWriterQos* method), 132
to_string () (*rti.connexdds.DomainParticipantFactoryQos* method), 165
to_string () (*rti.connexdds.DomainParticipantQos* method), 170
to_string () (*rti.connexdds.DynamicData* method), 244

- `to_string()` (*rti.connexdds.DynamicType* method), 255
`to_string()` (*rti.connexdds.PublisherQos* method), 476
`to_string()` (*rti.connexdds.SubscriberQos* method), 582
`to_string()` (*rti.connexdds.TopicQos* method), 685
`to_timedelta()` (*rti.connexdds.Duration* method), 184
`token_bucket` (*rti.connexdds.FlowControllerProperty* property), 281
`tokens_added_per_period` (*rti.connexdds.FlowControllerTokenBucketProperty* property), 285
`tokens_leaked_per_period` (*rti.connexdds.FlowControllerTokenBucketProperty* property), 285
`tolerance_source_timestamp_dropped_sample_count` (*rti.connexdds.DataReaderCacheStatus* property), 91
`Topic` (class in *rti.connexdds*), 638
`TOPIC` (*rti.connexdds.ActivityContextMask* attribute), 38
`topic` (*rti.connexdds.ContentFilteredTopic* property), 77
`topic` (*rti.connexdds.Data Writer* property), 118
`TOPIC` (*rti.connexdds.DestinationOrderScopeKind* attribute), 146
`TOPIC` (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind* attribute), 145
`topic` (*rti.connexdds.DynamicData.ContentFilteredTopic* property), 186
`topic` (*rti.connexdds.DynamicData.Data Writer* property), 202
`topic` (*rti.connexdds.ParticipantBuiltinTopicData.ContentFilteredTopic* property), 371
`topic` (*rti.connexdds.ParticipantBuiltinTopicData.Data Writer* property), 388
`TOPIC` (*rti.connexdds.PresentationAccessScopeKind* attribute), 417
`TOPIC` (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind* attribute), 416
`topic` (*rti.connexdds.PublicationBuiltinTopicData.ContentFilteredTopic* property), 427
`topic` (*rti.connexdds.PublicationBuiltinTopicData.Data Writer* property), 444
`topic` (*rti.connexdds.ServiceRequest.ContentFilteredTopic* property), 525
`topic` (*rti.connexdds.ServiceRequest.Data Writer* property), 541
`topic` (*rti.connexdds.SubscriptionBuiltinTopicData.ContentFilteredTopic* property), 586
`topic` (*rti.connexdds.SubscriptionBuiltinTopicData.Data Writer* property), 603
`topic` (*rti.connexdds.TopicBuiltinTopicData.ContentFilteredTopic* property), 641
`topic` (*rti.connexdds.TopicBuiltinTopicData.Data Writer* property), 658
`topic_access_scope()` (*rti.connexdds.Presentation* static method), 416
`topic_data` (*rti.connexdds.PublicationBuiltinTopicData* property), 463
`topic_data` (*rti.connexdds.SubscriptionBuiltinTopicData* property), 622
`topic_data` (*rti.connexdds.TopicBuiltinTopicData* property), 675
`topic_data` (*rti.connexdds.TopicQos* property), 685
`topic_data_max_length` (*rti.connexdds.DomainParticipantResourceLimits* property), 175
`topic_description` (*rti.connexdds.DataReader* property), 89
`topic_description` (*rti.connexdds.DynamicData.DataReader* property), 194
`topic_description` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* property), 379
`topic_description` (*rti.connexdds.PublicationBuiltinTopicData.DataReader* property), 435
`topic_description` (*rti.connexdds.ServiceRequest.DataReader* property), 533
`topic_description` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader* property), 594
`topic_description` (*rti.connexdds.TopicBuiltinTopicData.DataReader* property), 649
`topic_expression` (*rti.connexdds.MulticastMapping* property), 358
`topic_name` (*rti.connexdds.DataReader* property), 89
`topic_name` (*rti.connexdds.Data Writer* property), 118
`topic_name` (*rti.connexdds.DynamicData.DataReader* property), 194
`topic_name` (*rti.connexdds.DynamicData.Data Writer* property), 202
`topic_name` (*rti.connexdds.IAnyDataReader* property), 291
`topic_name` (*rti.connexdds.IAnyData Writer* property), 294
`topic_name` (*rti.connexdds.ParticipantBuiltinTopicData* attribute), 406
`topic_name` (*rti.connexdds.ParticipantBuiltinTopicData.DataReader* property), 379
`topic_name` (*rti.connexdds.ParticipantBuiltinTopicData.Data Writer* property), 388
`topic_name` (*rti.connexdds.PublicationBuiltinTopicData* property), 463
`topic_name` (*rti.connexdds.PublicationBuiltinTopicData.DataReader* property), 436
`topic_name` (*rti.connexdds.PublicationBuiltinTopicData.Data Writer* property), 444
`topic_name` (*rti.connexdds.ServiceRequest* attribute), 557
`topic_name` (*rti.connexdds.ServiceRequest.DataReader* property), 533

topic_name (*rti.connexdds.ServiceRequest.DataWriter* property), 541
 topic_name (*rti.connexdds.SubscriptionBuiltinTopicData* property), 622
 topic_name (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader* property), 594
 topic_name (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* property), 603
 topic_name (*rti.connexdds.TopicBuiltinTopicData* attribute), 675
 topic_name (*rti.connexdds.TopicBuiltinTopicData.DataReader* property), 649
 topic_name (*rti.connexdds.TopicBuiltinTopicData.DataWriter* property), 658
 topic_name (*rti.connexdds.TopicQueryData* property), 687
 topic_qos (*rti.connexdds.QosProvider* property), 484
 topic_qos_from_profile () (*rti.connexdds.QosProvider* method), 484
 TOPIC_QUERY (*rti.connexdds.ServiceRequestId* attribute), 559
 TOPIC_QUERY (*rti.connexdds.ServiceRequestId.ServiceRequestId* attribute), 559
 TOPIC_QUERY (*rti.connexdds.StreamKind* attribute), 569
 topic_query_dispatch (*rti.connexdds.DataWriterQos* property), 132
 topic_query_guid (*rti.connexdds.SampleInfo* property), 509
 topic_query_publication_thread (*rti.connexdds.AsynchronousPublisher* property), 52
 TopicBuiltinTopicData (class in *rti.connexdds*), 640
 TopicBuiltinTopicData.ContentFilter (class in *rti.connexdds*), 640
 TopicBuiltinTopicData.ContentFilteredTopic (class in *rti.connexdds*), 640
 TopicBuiltinTopicData.ContentFiltered-TopicSeq (class in *rti.connexdds*), 641
 TopicBuiltinTopicData.DataReader (class in *rti.connexdds*), 643
 TopicBuiltinTopicData.DataReaderListener (class in *rti.connexdds*), 649
 TopicBuiltinTopicData.DataReader.Selector (class in *rti.connexdds*), 643
 TopicBuiltinTopicData.DataReaderSeq (class in *rti.connexdds*), 650
 TopicBuiltinTopicData.DataWriter (class in *rti.connexdds*), 653
 TopicBuiltinTopicData.DataWriterListener (class in *rti.connexdds*), 660
 TopicBuiltinTopicData.DataWriterSeq (class in *rti.connexdds*), 662
 TopicBuiltinTopicData.ITopicDescription (class in *rti.connexdds*), 664
 TopicBuiltinTopicData.LoanedSample (class in *rti.connexdds*), 664
 TopicBuiltinTopicData.LoanedSamples (class in *rti.connexdds*), 665
 TopicBuiltinTopicData.NoOpDataReaderListener (class in *rti.connexdds*), 665
 TopicBuiltinTopicData.NoOpDataWriterListener (class in *rti.connexdds*), 666
 TopicBuiltinTopicData.NoOpTopicListener (class in *rti.connexdds*), 668
 TopicBuiltinTopicData.Sample (class in *rti.connexdds*), 668
 TopicBuiltinTopicDataSeq (class in *rti.connexdds*), 676
 TopicBuiltinTopicData.SharedSamples (class in *rti.connexdds*), 669
 TopicBuiltinTopicData.TimestampedSeq (class in *rti.connexdds*), 678
 TopicBuiltinTopicData.Topic (class in *rti.connexdds*), 669
 TopicBuiltinTopicData.TopicDescription (class in *rti.connexdds*), 670
 TopicBuiltinTopicData.TopicListener (class in *rti.connexdds*), 670
 TopicBuiltinTopicData.TopicSeq (class in *rti.connexdds*), 671
 TopicBuiltinTopicData.ValidLoanedSamples (class in *rti.connexdds*), 673
 TopicBuiltinTopicData.WriterContentFilter (class in *rti.connexdds*), 673
 TopicBuiltinTopicData.WriterContentFilterHelper (class in *rti.connexdds*), 674
 TopicData (class in *rti.connexdds*), 680
 TopicDescription (class in *rti.connexdds*), 681
 TopicListener (class in *rti.connexdds*), 681
 TopicQos (class in *rti.connexdds*), 682
 TopicQuery (class in *rti.connexdds*), 686
 TopicQueryData (class in *rti.connexdds*), 687
 TopicQueryDispatch (class in *rti.connexdds*), 687
 TopicQuerySelection (class in *rti.connexdds*), 688
 TopicQuerySelectionKind (class in *rti.connexdds*), 688
 TopicQuerySelectionKind.TopicQuerySelectionKind (class in *rti.connexdds*), 688
 TopicSeq (class in *rti.connexdds*), 690
 total (*rti.connexdds.EventCount32* property), 267
 total (*rti.connexdds.EventCount64* property), 267
 total_count (*rti.connexdds.InconsistentTopicStatus* property), 308
 total_count (*rti.connexdds.LivelinessLostStatus* property), 330
 total_count (*rti.connexdds.OfferedDeadlineMissedStatus* property), 366
 total_count (*rti.connexdds.OfferedIncompatibleQosStatus* property), 366
 total_count (*rti.connexdds.PublicationMatchedStatus* property), 469

total_count (*rti.connexdds.RequestedDeadlineMissedStatus property*), 496

total_count (*rti.connexdds.SampleLostStatus property*), 513

total_count (*rti.connexdds.SampleRejectedStatus property*), 516

total_count (*rti.connexdds.ServiceRequestAcceptedStatus property*), 558

total_count (*rti.connexdds.SubscriptionMatchedStatus property*), 628

total_count () (*rti.connexdds.RequestedIncompatibleQosStatus method*), 496

total_count_change (*rti.connexdds.InconsistentTopicStatus property*), 308

total_count_change (*rti.connexdds.LivelinessLostStatus property*), 331

total_count_change (*rti.connexdds.OfferedDeadlineMissedStatus property*), 366

total_count_change (*rti.connexdds.OfferedIncompatibleQosStatus property*), 366

total_count_change (*rti.connexdds.PublicationMatchedStatus property*), 469

total_count_change (*rti.connexdds.RequestedDeadlineMissedStatus property*), 496

total_count_change (*rti.connexdds.RequestedIncompatibleQosStatus property*), 496

total_count_change (*rti.connexdds.SampleLostStatus property*), 513

total_count_change (*rti.connexdds.SampleRejectedStatus property*), 516

total_count_change (*rti.connexdds.SubscriptionMatchedStatus property*), 628

total_element_count (*rti.connexdds.ArrayType property*), 51

total_samples_dropped_by_instance_replacement (*rti.connexdds.DataReaderCacheStatus property*), 91

TRACE (*rti.logging.distlog.LogLevel attribute*), 771

trace () (*rti.logging.distlog.Logger static method*), 772

trace_file_name (*rti.connexdds.PersistentStorageSettings property*), 413

transient (*rti.connexdds.Durability attribute*), 179

TRANSIENT (*rti.connexdds.DurabilityKind attribute*), 180

TRANSIENT (*rti.connexdds.DurabilityKind.DurabilityKind attribute*), 180

transient_local (*rti.connexdds.Durability attribute*), 179

TRANSIENT_LOCAL (*rti.connexdds.DurabilityKind attribute*), 180

TRANSIENT_LOCAL (*rti.connexdds.DurabilityKind.DurabilityKind attribute*), 180

transport_builtin (*rti.connexdds.DomainParticipantQos property*), 171

transport_info (*rti.connexdds.ParticipantBuiltinTopicData property*), 406

transport_info_list_max_length (*rti.connexdds.DomainParticipantResourceLimits property*), 175

transport_multicast (*rti.connexdds.DataReaderQos property*), 103

transport_multicast_mapping (*rti.connexdds.DomainParticipantQos property*), 171

transport_priority (*rti.connexdds.DataReaderQos property*), 103

transport_priority (*rti.connexdds.DataWriterQos property*), 132

transport_priority (*rti.connexdds.TopicBuiltinTopicData property*), 675

transport_priority (*rti.connexdds.TopicQos property*), 686

transport_selection (*rti.connexdds.DataReaderQos property*), 103

transport_selection (*rti.connexdds.DataWriterQos property*), 132

transport_unicast (*rti.connexdds.DataReaderQos property*), 103

transport_unicast (*rti.connexdds.DataWriterQos property*), 132

TransportBuiltin (*class in rti.connexdds*), 692

TransportBuiltinMask (*class in rti.connexdds*), 693

TransportClassId (*class in rti.connexdds*), 696

TransportClassId.TransportClassId (*class in rti.connexdds*), 696

TransportInfo (*class in rti.connexdds*), 699

TransportInfoSeq (*class in rti.connexdds*), 699

TransportInfoVector (*class in rti.connexdds*), 701

TransportMulticast (*class in rti.connexdds*), 702

TransportMulticastKind (*class in rti.connexdds*), 703

TransportMulticastKind.TransportMulticastKind (*class in rti.connexdds*), 703

TransportMulticastMapping (*class in rti.connexdds*), 705

TransportMulticastMappingFunction (*class in rti.connexdds*), 705

TransportMulticastSeq (*class in rti.connexdds*), 706

TransportMulticastSettings (*class in rti.connexdds*), 708

TransportMulticastSettingsSeq (*class in rti.connexdds*), 709

TransportPriority (*class in rti.connexdds*), 711

transports (*rti.connexdds.TransportMulticastSettings property*), 708

transports (*rti.connexdds.TransportUnicastSettings property*), 713

TransportSelection (*class in rti.connexdds*), 711

TransportUnicast (*class in rti.connexdds*), 712

TransportUnicastSettings (*class in rti.connexdds*), 713

TransportUnicastSettingsSeq (*class in rti.connexdds*), 713

trigger_flow() (rti.connexdds.FlowController method), 280
 trigger_value (rti.connexdds.GuardCondition property), 287
 trigger_value (rti.connexdds.ICondition property), 299
 trigger_value (rti.connexdds.StatusCondition property), 566
 TriggeredConditions (class in rti.connexdds), 716
 TriggeredConditionsIterator (class in rti.connexdds), 716
 trim_to_size (rti.connexdds.DynamicDataTypeSerializationProperty property), 255
 TRUNCATE (rti.connexdds.PersistentJournalKind attribute), 412
 try_get() (rti.connexdds.DataTag method), 114
 try_get() (rti.connexdds.Property method), 425
 TYPE (rti.connexdds.ActivityContextMask attribute), 38
 type (rti.connexdds.DynamicData property), 245
 type (rti.connexdds.Member property), 347
 type (rti.connexdds.PublicationBuiltinTopicData property), 463
 type (rti.connexdds.SubscriptionBuiltinTopicData property), 622
 type (rti.connexdds.Topic property), 639
 type (rti.connexdds.UnionMember property), 738
 type() (rti.connexdds.QosProvider method), 484
 type_code_max_serialized_length (rti.connexdds.DomainParticipantResourceLimits property), 175
 type_consistency (rti.connexdds.DataReaderQos property), 103
 type_consistency_enforcement (rti.connexdds.DataReaderQos property), 104
 type_kind (rti.connexdds.DynamicData property), 245
 type_libraries (rti.connexdds.QosProvider property), 484
 type_name (rti.connexdds.DataReader property), 89
 type_name (rti.connexdds.DataWriter property), 118
 type_name (rti.connexdds.DynamicData.DataReader property), 194
 type_name (rti.connexdds.DynamicData.DataWriter property), 202
 type_name (rti.connexdds.DynamicData.ITopicDescription property), 209
 type_name (rti.connexdds.IAnyDataReader property), 291
 type_name (rti.connexdds.IAnyDataWriter property), 294
 type_name (rti.connexdds.IAnyTopic property), 297
 type_name (rti.connexdds.ITopicDescription property), 305
 type_name (rti.connexdds.ParticipantBuiltinTopicData.DataReader property), 379
 type_name (rti.connexdds.ParticipantBuiltinTopicData.DataWriter property), 388
 type_name (rti.connexdds.ParticipantBuiltinTopicData.ITopicDescription property), 395
 type_name (rti.connexdds.PublicationBuiltinTopicData property), 463
 type_name (rti.connexdds.PublicationBuiltinTopicData.DataReader property), 436
 type_name (rti.connexdds.PublicationBuiltinTopicData.DataWriter property), 444
 type_name (rti.connexdds.PublicationBuiltinTopicData.ITopicDescription property), 451
 type_name (rti.connexdds.ServiceRequest.DataReader property), 533
 type_name (rti.connexdds.ServiceRequest.DataWriter property), 541
 type_name (rti.connexdds.ServiceRequest.ITopicDescription property), 547
 type_name (rti.connexdds.SubscriptionBuiltinTopicData property), 622
 type_name (rti.connexdds.SubscriptionBuiltinTopicData.DataReader property), 594
 type_name (rti.connexdds.SubscriptionBuiltinTopicData.DataWriter property), 603
 type_name (rti.connexdds.SubscriptionBuiltinTopicData.ITopicDescription property), 610
 type_name (rti.connexdds.TopicBuiltinTopicData property), 676
 type_name (rti.connexdds.TopicBuiltinTopicData.DataReader property), 649
 type_name (rti.connexdds.TopicBuiltinTopicData.DataWriter property), 658
 type_name (rti.connexdds.TopicBuiltinTopicData.ITopicDescription property), 664
 type_object_max_deserialized_length (rti.connexdds.DomainParticipantResourceLimits property), 175
 type_object_max_serialized_length (rti.connexdds.DomainParticipantResourceLimits property), 175
 type_support (rti.connexdds.DataReaderQos property), 104
 type_support (rti.connexdds.DataWriterQos property), 132
 type_support (rti.connexdds.Topic property), 639
 TypeConsistencyEnforcement (class in rti.connexdds), 716
 TypeConsistencyEnforcementKind (class in rti.connexdds), 717
 TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind (class in rti.connexdds), 717
 TypeKind (class in rti.connexdds), 720
 TypeKind.TypeKind (class in rti.connexdds), 721
 TypeSupport (class in rti.connexdds), 725

U

UDPv4 (rti.connexdds.LocatorKind attribute), 337
 UDPv4 (rti.connexdds.LocatorKind.LocatorKind attribute), 336
 udpv4 (rti.connexdds.TransportBuiltin attribute), 693
 UDPv4 (rti.connexdds.TransportBuiltinMask attribute), 693

- UDPv4 (*rti.connexdds.TransportClassId* attribute), 698
- UDPv4 (*rti.connexdds.TransportClassId.TransportClassId* attribute), 697
- UDPv4_WAN (*rti.connexdds.TransportClassId* attribute), 698
- UDPv4_WAN (*rti.connexdds.TransportClassId.TransportClassId* attribute), 697
- UDPv6 (*rti.connexdds.LocatorKind* attribute), 337
- UDPv6 (*rti.connexdds.LocatorKind.LocatorKind* attribute), 336
- udpv6 (*rti.connexdds.TransportBuiltin* attribute), 693
- UDPv6 (*rti.connexdds.TransportBuiltinMask* attribute), 693
- UDPv6 (*rti.connexdds.TransportClassId* attribute), 698
- UDPv6 (*rti.connexdds.TransportClassId.TransportClassId* attribute), 697
- UDPv6_510 (*rti.connexdds.LocatorKind* attribute), 337
- UDPv6_510 (*rti.connexdds.LocatorKind.LocatorKind* attribute), 336
- UDPv6_510 (*rti.connexdds.TransportClassId* attribute), 698
- UDPv6_510 (*rti.connexdds.TransportClassId.TransportClassId* attribute), 697
- UInt8Seq (class in *rti.connexdds*), 734
- UInt8Type (class in *rti.connexdds*), 736
- UInt16Seq (class in *rti.connexdds*), 726
- UInt16Type (class in *rti.connexdds*), 728
- UInt32Seq (class in *rti.connexdds*), 729
- UInt32Type (class in *rti.connexdds*), 731
- UInt64Seq (class in *rti.connexdds*), 731
- UInt64Type (class in *rti.connexdds*), 734
- UINT_8_TYPE (*rti.connexdds.TypeKind* attribute), 725
- UINT_8_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 723
- UINT_16_TYPE (*rti.connexdds.TypeKind* attribute), 724
- UINT_16_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 723
- UINT_32_TYPE (*rti.connexdds.TypeKind* attribute), 724
- UINT_32_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 723
- UINT_64_TYPE (*rti.connexdds.TypeKind* attribute), 724
- UINT_64_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 723
- ULongLongSeq (in module *rti.connexdds*), 726
- ULongLongType (in module *rti.connexdds*), 726
- ULongSeq (in module *rti.connexdds*), 726
- ULongType (in module *rti.connexdds*), 726
- unacknowledged_sample_count (*rti.connexdds.ReliableWriterCacheChangedStatus* property), 493
- unacknowledged_sample_count_peak (*rti.connexdds.ReliableWriterCacheChangedStatus* property), 493
- UNBOUNDED (*rti.connexdds.UnidimensionalCollectionType* attribute), 736
- uncommitted_sample_count (*rti.connexdds.DataReaderProtocolStatus* property), 97
- underlying (*rti.connexdds.AcknowledgmentKind* property), 37
- underlying (*rti.connexdds.CdrPaddingKind* property), 63
- underlying (*rti.connexdds.DataReaderInstanceRemovalKind* property), 94
- underlying (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind* property), 137
- underlying (*rti.connexdds.DestinationOrderKind* property), 144
- underlying (*rti.connexdds.DestinationOrderScopeKind* property), 146
- underlying (*rti.connexdds.DurabilityKind* property), 181
- underlying (*rti.connexdds.DynamicDataEncapsulationKind* property), 247
- underlying (*rti.connexdds.ExtensibilityKind* property), 271
- underlying (*rti.connexdds.FlowControllerSchedulingPolicy* property), 284
- underlying (*rti.connexdds.HistoryKind* property), 290
- underlying (*rti.connexdds.IgnoredEntityReplacementKind* property), 307
- underlying (*rti.connexdds.LivelinessKind* property), 330
- underlying (*rti.connexdds.LocatorKind* property), 338
- underlying (*rti.connexdds.LogCategory* property), 344
- underlying (*rti.connexdds.OwnershipKind* property), 369
- underlying (*rti.connexdds.PresentationAccessScopeKind* property), 418
- underlying (*rti.connexdds.PrintFormat* property), 420
- underlying (*rti.connexdds.PrintFormatKind* property), 422
- underlying (*rti.connexdds.PublishModeKind* property), 472
- underlying (*rti.connexdds.ReliabilityKind* property), 492
- underlying (*rti.connexdds.RemoteParticipantPurgeKind* property), 495
- underlying (*rti.connexdds.ServiceKind* property), 523
- underlying (*rti.connexdds.ServiceRequestId* property), 560
- underlying (*rti.connexdds.ThreadSettingsCpuRotationKind* property), 633
- underlying (*rti.connexdds.TopicQuerySelectionKind* property), 690
- underlying (*rti.connexdds.TransportClassId* property), 699
- underlying (*rti.connexdds.TransportMulticastKind* property), 705
- underlying (*rti.connexdds.TypeConsistencyEnforcementKind* property), 720
- underlying (*rti.connexdds.TypeKind* property), 725
- underlying (*rti.connexdds.Verbosity* property), 744
- underlying (*rti.connexdds.WireProtocolAutoKind* property), 755
- UNICAST (*rti.connexdds.TransportMulticastKind* attribute), 704
- UNICAST (*rti.connexdds.TransportMulticastKind.TransportMulticastKind* attribute), 703
- unicast_locators (*rti.connexdds.PublicationBuiltinTopicData* property), 463
- unicast_locators (*rti.connexdds.SubscriptionBuiltinTopicData* property), 622
- UnidimensionalCollectionType (class in *rti.connexdds*), 736
- UNION_FWD_DECL_TYPE (*rti.connexdds.TypeKind* attribute), 725
- UNION_FWD_DECL_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 723

- UNION_TYPE (*rti.connexdds.TypeKind* attribute), 725
 - UNION_TYPE (*rti.connexdds.TypeKind.TypeKind* attribute), 723
 - UnionMember (*class in rti.connexdds*), 737
 - UnionMemberSeq (*class in rti.connexdds*), 738
 - UnionType (*class in rti.connexdds*), 740
 - UNKNOWN (*rti.connexdds.CoherentSetInfo* attribute), 68
 - unknown (*rti.connexdds.Guid* attribute), 288
 - unknown (*rti.connexdds.ProductVersion* attribute), 423
 - unknown (*rti.connexdds.SampleIdentity* attribute), 508
 - unknown (*rti.connexdds.SequenceNumber* attribute), 520
 - UNKNOWN (*rti.connexdds.ServiceRequestId* attribute), 559
 - UNKNOWN (*rti.connexdds.ServiceRequestId.ServiceRequestId* attribute), 559
 - unknown (*rti.connexdds.VendorId* attribute), 742
 - unload_profiles () (*rti.connexdds.QosProvider* method), 484
 - unregister_contentfilter () (*rti.connexdds.DomainParticipant* method), 163
 - unregister_instance () (*rti.connexdds.DataWriter* method), 118
 - unregister_instance () (*rti.connexdds.DynamicData.DataWriter* method), 202
 - unregister_instance () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter* method), 388
 - unregister_instance () (*rti.connexdds.PublicationBuiltinTopicData.DataWriter* method), 444
 - unregister_instance () (*rti.connexdds.ServiceRequest.DataWriter* method), 541
 - unregister_instance () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* method), 603
 - unregister_instance () (*rti.connexdds.TopicBuiltinTopicData.DataWriter* method), 658
 - unregister_instance_async () (*rti.connexdds.DynamicData.DataWriter* method), 202
 - unregister_instance_async () (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter* method), 388
 - unregister_instance_async () (*rti.connexdds.PublicationBuiltinTopicData.DataWriter* method), 445
 - unregister_instance_async () (*rti.connexdds.ServiceRequest.DataWriter* method), 541
 - unregister_instance_async () (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter* method), 603
 - unregister_instance_async () (*rti.connexdds.TopicBuiltinTopicData.DataWriter* method), 658
 - unregister_thread () (*in module rti.connexdds*), 759
 - unregister_type () (*rti.connexdds.DomainParticipant* method), 163
 - UNREGISTERED (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind* attribute), 136
 - UNREGISTERED (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind* attribute), 135
 - unregistered_instance_count (*rti.connexdds.DataWriterCacheStatus* property), 120
 - unregistered_instance_count_peak (*rti.connexdds.DataWriterCacheStatus* property), 120
 - unretain () (*rti.connexdds.IEntity* method), 302
 - unretain () (*rti.connexdds.TopicQuery* method), 686
 - UnsupportedError, 741
 - update () (*rti.connexdds.DynamicData* method), 245
 - url_profile (*rti.connexdds.QosProviderParams* property), 485
 - use_count (*rti.connexdds.IEntity* property), 302
 - use_reader_content_filter () (*rti.connexdds.TopicQuery* static method), 686
 - use_shared_exclusive_area (*rti.connexdds.ExclusiveArea* property), 268
 - user (*rti.connexdds.LogCategory* attribute), 344
 - user (*rti.connexdds.LogCategory.LogCategory* attribute), 343
 - user_data (*rti.connexdds.DataReaderQos* property), 104
 - user_data (*rti.connexdds.DataWriterQos* property), 132
 - user_data (*rti.connexdds.DomainParticipantQos* property), 171
 - user_data (*rti.connexdds.ParticipantBuiltinTopicData* property), 406
 - user_data (*rti.connexdds.PublicationBuiltinTopicData* property), 464
 - user_data (*rti.connexdds.SubscriptionBuiltinTopicData* property), 622
 - user_forwarding_level (*rti.connexdds.MonitoringLoggingForwardingSettings* property), 353
 - USER_MULTICAST (*rti.connexdds.RtpsReservedPortKindMask* attribute), 500
 - user_multicast_port_offset (*rti.connexdds.RtpsWellKnownPorts* property), 504
 - USER_UNICAST (*rti.connexdds.RtpsReservedPortKindMask* attribute), 500
 - user_unicast_port_offset (*rti.connexdds.RtpsWellKnownPorts* property), 504
 - UserData (*class in rti.connexdds*), 741
 - UserDataSample (*class in rti.connexdds*), 741
 - UshortSeq (*in module rti.connexdds*), 741
 - UShortType (*in module rti.connexdds*), 726
- ## V
- vacuum (*rti.connexdds.PersistentStorageSettings* property), 413
 - valid (*rti.connexdds.SampleInfo* property), 509
 - valid_response_data (*rti.connexdds.AcknowledgmentInfo* property), 35
 - value (*rti.connexdds.AcknowledgmentKind.AcknowledgmentKind* property), 36

- value (*rti.connexdds.AckResponseData* property), 34
- value (*rti.connexdds.BuiltinTopicKey* property), 59
- value (*rti.connexdds.CdrPaddingKind.CdrPaddingKind* property), 62
- value (*rti.connexdds.Cookie* property), 80
- value (*rti.connexdds.DataReaderInstanceRemovalKind.DataReaderInstanceRemovalKind* property), 93
- value (*rti.connexdds.DataRepresentation* property), 110
- value (*rti.connexdds.DataWriterResourceLimitsInstanceReplacementKind.DataWriterResourceLimitsInstanceReplacementKind* property), 136
- value (*rti.connexdds.DestinationOrderKind.DestinationOrderKind* property), 143
- value (*rti.connexdds.DestinationOrderScopeKind.DestinationOrderScopeKind* property), 145
- value (*rti.connexdds.DurabilityKind.DurabilityKind* property), 180
- value (*rti.connexdds.DynamicDataEncapsulationKind.DynamicDataEncapsulationKind* property), 246
- value (*rti.connexdds.ExtensibilityKind.ExtensibilityKind* property), 270
- value (*rti.connexdds.FlowControllerSchedulingPolicy.FlowControllerSchedulingPolicy* property), 283
- value (*rti.connexdds.GroupData* property), 286
- value (*rti.connexdds.HistoryKind.HistoryKind* property), 289
- value (*rti.connexdds.IgnoredEntityReplacementKind.IgnoredEntityReplacementKind* property), 306
- value (*rti.connexdds.InstanceStateConsistencyKind* property), 314
- value (*rti.connexdds.LivelinessKind.LivelinessKind* property), 329
- value (*rti.connexdds.LocatorKind.LocatorKind* property), 337
- value (*rti.connexdds.LogCategory.LogCategory* property), 343
- value (*rti.connexdds.OwnershipKind.OwnershipKind* property), 368
- value (*rti.connexdds.OwnershipStrength* property), 369
- value (*rti.connexdds.PersistentJournalKind* property), 413
- value (*rti.connexdds.PersistentSynchronizationKind* property), 415
- value (*rti.connexdds.PresentationAccessScopeKind.PresentationAccessScopeKind* property), 417
- value (*rti.connexdds.PrintFormatKind.PrintFormatKind* property), 421
- value (*rti.connexdds.PrintFormat.PrintFormat* property), 419
- value (*rti.connexdds.PublishModeKind.PublishModeKind* property), 471
- value (*rti.connexdds.ReliabilityKind.ReliabilityKind* property), 491
- value (*rti.connexdds.RemoteParticipantPurgeKind.RemoteParticipantPurgeKind* property), 494
- value (*rti.connexdds.SequenceNumber* property), 520
- value (*rti.connexdds.ServiceKind.ServiceKind* property), 523
- value (*rti.connexdds.ServiceRequestId.ServiceRequestId* property), 559
- value (*rti.connexdds.SyslogVerbosity* property), 629
- value (*rti.connexdds.ThreadSettingsCpuRotationKind.ThreadSettingsCpuRotationKind* property), 632
- value (*rti.connexdds.TopicData* property), 681
- value (*rti.connexdds.TopicQuerySelectionKind.TopicQuerySelectionKind* property), 689
- value (*rti.connexdds.TransportClassId.TransportClassId* property), 698
- value (*rti.connexdds.TransportMulticast* property), 703
- value (*rti.connexdds.TransportMulticastKind.TransportMulticastKind* property), 704
- value (*rti.connexdds.TransportMulticastMapping* property), 705
- value (*rti.connexdds.TransportPriority* property), 711
- value (*rti.connexdds.TransportSelection* property), 712
- value (*rti.connexdds.TransportUnicast* property), 712
- value (*rti.connexdds.TypeConsistencyEnforcementKind.TypeConsistencyEnforcementKind* property), 719
- value (*rti.connexdds.TypeKind.TypeKind* property), 724
- value (*rti.connexdds.UserData* property), 741
- value (*rti.connexdds.VendorId* property), 742
- value (*rti.connexdds.Verbosity.Verbosity* property), 743
- value (*rti.connexdds.WireProtocolAutoKind.WireProtocolAutoKind* property), 754
- value (*rti.logging.distlog.LogLevel* property), 771
- value (*rti.types.builtin.Bytes* attribute), 759
- value (*rti.types.builtin.KeyedBytes* attribute), 759
- value (*rti.types.builtin.KeyedString* attribute), 760
- value (*rti.types.builtin.String* attribute), 760
- values () (*rti.connexdds.StringMap* method), 572
- vendor_builtin_endpoints (*rti.connexdds.ParticipantBuiltinTopicData* property), 406
- VendorId (class in *rti.connexdds*), 742
- VERBOSE (*rti.connexdds.PrintFormat* attribute), 420
- VERBOSE (*rti.connexdds.PrintFormat.PrintFormat* attribute), 419
- VERBOSE_TIMESTAMPED (*rti.connexdds.PrintFormat* attribute), 420
- VERBOSE_TIMESTAMPED (*rti.connexdds.PrintFormat.PrintFormat* attribute), 419
- Verbosity (class in *rti.connexdds*), 742
- verbosity (*rti.connexdds.Logger* property), 345
- verbosity () (*rti.logging.distlog.Logger* static method), 772
- verbosity_by_category () (*rti.connexdds.Logger* method), 345
- Verbosity.Verbosity (class in *rti.connexdds*), 742
- view_state (*rti.connexdds.DataState* property), 111
- view_state (*rti.connexdds.DataStateEx* property), 113

ViewState (class in *rti.connextdds*), 744
 virtual_duplicate_dropped_sample_count
 (*rti.connextdds.DataReaderCacheStatus* property), 91
 virtual_guid (*rti.connextdds.DataReaderProtocol* property),
 95
 virtual_guid (*rti.connextdds.DataWriterProtocol* property),
 122
 virtual_guid (*rti.connextdds.PublicationBuiltinTopicData*
 property), 464
 virtual_guid (*rti.connextdds.SubscriptionBuiltinTopicData*
 property), 622
 virtual_heartbeat_period
 (*rti.connextdds.RtpsReliableWriterProtocol* property),
 500
 volatile (*rti.connextdds.Durability* attribute), 179
 VOLATILE (*rti.connextdds.DurabilityKind* attribute), 180
 VOLATILE (*rti.connextdds.DurabilityKind.DurabilityKind*
 attribute), 180

W

wait () (*rti.connextdds.WaitSet* method), 748
 wait_async () (*rti.connextdds.WaitSet* method), 748
 wait_for_acknowledgments ()
 (*rti.connextdds.DataWriter* method), 119
 wait_for_acknowledgments ()
 (*rti.connextdds.DynamicData.DataWriter* method),
 203
 wait_for_acknowledgments ()
 (*rti.connextdds.IAnyDataWriter* method), 294
 wait_for_acknowledgments () (*rti.connextdds.Partici-*
 pantBuiltinTopicData.DataWriter method),
 389
 wait_for_acknowledgments () (*rti.connextdds.Publica-*
 tionBuiltinTopicData.DataWriter method),
 445
 wait_for_acknowledgments () (*rti.connextdds.Publisher*
 method), 473
 wait_for_acknowledgments ()
 (*rti.connextdds.ServiceRequest.DataWriter* method),
 542
 wait_for_acknowledgments () (*rti.connextdds.Subscrip-*
 tionBuiltinTopicData.DataWriter method),
 604
 wait_for_acknowledgments ()
 (*rti.connextdds.TopicBuiltinTopicData.DataWriter*
 method), 658
 wait_for_asynchronous_publishing ()
 (*rti.connextdds.DataWriter* method), 119
 wait_for_asynchronous_publishing ()
 (*rti.connextdds.DynamicData.DataWriter* method),
 203
 wait_for_asynchronous_publishing () (*rti.con-*
 nextdds.ParticipantBuiltinTopicData.DataWriter
 method), 389
 wait_for_asynchronous_publishing () (*rti.con-*
 nextdds.PublicationBuiltinTopicData.DataWriter
 method), 445
 wait_for_asynchronous_publishing ()
 (*rti.connextdds.Publisher* method), 473
 wait_for_asynchronous_publishing ()
 (*rti.connextdds.ServiceRequest.DataWriter* method),
 542
 wait_for_asynchronous_publishing () (*rti.con-*
 nextdds.SubscriptionBuiltinTopicData.DataWriter
 method), 604
 wait_for_asynchronous_publishing ()
 (*rti.connextdds.TopicBuiltinTopicData.DataWriter*
 method), 659
 wait_for_asynchronous_publishing_async ()
 (*rti.connextdds.DataWriter* method), 119
 wait_for_asynchronous_publishing_async ()
 (*rti.connextdds.DynamicData.DataWriter* method),
 203
 wait_for_asynchronous_publishing_async ()
 (*rti.connextdds.ParticipantBuiltinTopic-*
 Data.DataWriter method),
 389
 wait_for_asynchronous_publishing_async ()
 (*rti.connextdds.PublicationBuiltinTopic-*
 Data.DataWriter method),
 446
 wait_for_asynchronous_publishing_async ()
 (*rti.connextdds.ServiceRequest.DataWriter* method),
 542
 wait_for_asynchronous_publishing_async ()
 (*rti.connextdds.SubscriptionBuiltinTopic-*
 Data.DataWriter method),
 604
 wait_for_asynchronous_publishing_async ()
 (*rti.connextdds.TopicBuiltinTopicData.DataWriter*
 method), 659
 wait_for_historical_data ()
 (*rti.connextdds.DataReader* method), 90
 wait_for_historical_data ()
 (*rti.connextdds.DynamicData.DataReader* method),
 194
 wait_for_historical_data () (*rti.connextdds.Partici-*
 pantBuiltinTopicData.DataReader method),
 379
 wait_for_historical_data () (*rti.connextdds.Publica-*
 tionBuiltinTopicData.DataReader method),
 436
 wait_for_historical_data ()
 (*rti.connextdds.ServiceRequest.DataReader* method),
 533
 wait_for_historical_data () (*rti.connextdds.Subscrip-*
 tionBuiltinTopicData.DataReader method),
 594
 wait_for_historical_data ()
 (*rti.connextdds.TopicBuiltinTopicData.DataReader*
 method), 649
 wait_for_historical_data_async ()
 (*rti.connextdds.DataReader* method), 90
 wait_for_historical_data_async ()
 (*rti.connextdds.DynamicData.DataReader* method),
 194
 wait_for_historical_data_async () (*rti.con-*
 nextdds.ParticipantBuiltinTopicData.DataReader

- method*), 379
- `wait_for_historical_data_async()` (*rti.connexdds.PublicationBuiltinTopicData.DataReader method*), 436
- `wait_for_historical_data_async()` (*rti.connexdds.ServiceRequest.DataReader method*), 533
- `wait_for_historical_data_async()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataReader method*), 594
- `wait_for_historical_data_async()` (*rti.connexdds.TopicBuiltinTopicData.DataReader method*), 649
- `wait_for_replies()` (*rti.rpc.Requester method*), 763
- `wait_for_replies_async()` (*rti.rpc.Requester method*), 763
- `wait_for_requests()` (*rti.rpc.Replier method*), 766
- `wait_for_requests_async()` (*rti.rpc.Replier method*), 767
- `WaitSet` (*class in rti.connexdds*), 747
- `WaitSetProperty` (*class in rti.connexdds*), 749
- `WAL` (*rti.connexdds.PersistentJournalKind attribute*), 412
- `WARNING` (*rti.connexdds.SyslogVerbosity attribute*), 629
- `WARNING` (*rti.connexdds.Verbosity attribute*), 743
- `WARNING` (*rti.connexdds.Verbosity.Verbosity attribute*), 743
- `WARNING` (*rti.logging.distlog.LogLevel attribute*), 771
- `warning()` (*rti.connexdds.Logger method*), 345
- `warning()` (*rti.logging.distlog.Logger static method*), 772
- `WcharSeq` (*class in rti.connexdds*), 749
- `WcharType` (*class in rti.connexdds*), 751
- `WEB_INTEGRATION` (*rti.connexdds.ServiceKind attribute*), 523
- `WEB_INTEGRATION` (*rti.connexdds.ServiceKind.ServiceKind attribute*), 522
- `wire_protocol` (*rti.connexdds.DomainParticipantQos property*), 171
- `WireProtocol` (*class in rti.connexdds*), 752
- `WireProtocolAutoKind` (*class in rti.connexdds*), 753
- `WireProtocolAutoKind.WireProtocolAutoKind` (*class in rti.connexdds*), 753
- `write()` (*rti.connexdds.DataWriter method*), 119
- `write()` (*rti.connexdds.DynamicData.DataWriter method*), 203
- `write()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*), 389
- `write()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*), 446
- `write()` (*rti.connexdds.ServiceRequest.DataWriter method*), 542
- `write()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method*), 604
- `write()` (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 659
- `write_async()` (*rti.connexdds.DynamicData.DataWriter method*), 204
- `write_async()` (*rti.connexdds.ParticipantBuiltinTopicData.DataWriter method*), 390
- `write_async()` (*rti.connexdds.PublicationBuiltinTopicData.DataWriter method*), 446
- `write_async()` (*rti.connexdds.ServiceRequest.DataWriter method*), 543
- `write_async()` (*rti.connexdds.SubscriptionBuiltinTopicData.DataWriter method*), 605
- `write_async()` (*rti.connexdds.TopicBuiltinTopicData.DataWriter method*), 660
- `WriteParams` (*class in rti.connexdds*), 755
- `writer_attach()` (*rti.connexdds.DynamicData.WriterContentFilter method*), 219
- `writer_attach()` (*rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 404
- `writer_attach()` (*rti.connexdds.PublicationBuiltinTopicData.WriterContentFilter method*), 460
- `writer_attach()` (*rti.connexdds.ServiceRequest.WriterContentFilter method*), 556
- `writer_attach()` (*rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 619
- `writer_attach()` (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilter method*), 673
- `writer_compile()` (*rti.connexdds.DynamicData.WriterContentFilter method*), 219
- `writer_compile()` (*rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 404
- `writer_compile()` (*rti.connexdds.PublicationBuiltinTopicData.WriterContentFilter method*), 460
- `writer_compile()` (*rti.connexdds.ServiceRequest.WriterContentFilter method*), 556
- `writer_compile()` (*rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 619
- `writer_compile()` (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilter method*), 673
- `writer_compression_level` (*rti.connexdds.CompressionSettings property*), 73
- `writer_compression_threshold` (*rti.connexdds.CompressionSettings property*), 73
- `writer_data_lifecycle` (*rti.connexdds.DataWriterQos property*), 132
- `writer_data_tag_list_max_length` (*rti.connexdds.DomainParticipantResourceLimits property*), 175
- `writer_data_tag_string_max_length`

- `(rti.connexdds.DomainParticipantResourceLimits property)`, 175
- `writer_depth (rti.connexdds.Durability property)`, 179
- `writer_detach ()` (*rti.connexdds.DynamicData.WriterContentFilter method*), 219
- `writer_detach ()` (*rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 404
- `writer_detach ()` (*rti.connexdds.PublicationBuiltinTopicData.WriterContentFilter method*), 461
- `writer_detach ()` (*rti.connexdds.ServiceRequest.WriterContentFilter method*), 556
- `writer_detach ()` (*rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 619
- `writer_detach ()` (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilter method*), 673
- `writer_evaluate ()` (*rti.connexdds.DynamicData.WriterContentFilter method*), 219
- `writer_evaluate ()` (*rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 404
- `writer_evaluate ()` (*rti.connexdds.PublicationBuiltinTopicData.WriterContentFilter method*), 461
- `writer_evaluate ()` (*rti.connexdds.ServiceRequest.WriterContentFilter method*), 556
- `writer_evaluate ()` (*rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 619
- `writer_evaluate ()` (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilter method*), 674
- `writer_evaluate_helper ()` (*rti.connexdds.DynamicData.WriterContentFilterHelper method*), 220
- `writer_evaluate_helper ()` (*rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilterHelper method*), 405
- `writer_evaluate_helper ()` (*rti.connexdds.PublicationBuiltinTopicData.WriterContentFilterHelper method*), 461
- `writer_evaluate_helper ()` (*rti.connexdds.ServiceRequest.WriterContentFilterHelper method*), 557
- `writer_evaluate_helper ()` (*rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilterHelper method*), 620
- `writer_evaluate_helper ()` (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilterHelper method*), 674
- `writer_finalize ()` (*rti.connexdds.DynamicData.WriterContentFilter method*), 219
- `writer_finalize ()` (*rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 404
- `writer_finalize ()` (*rti.connexdds.PublicationBuiltinTopicData.WriterContentFilter method*), 461
- `writer_finalize ()` (*rti.connexdds.ServiceRequest.WriterContentFilter method*), 556
- `writer_finalize ()` (*rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 619
- `writer_finalize ()` (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilter method*), 674
- `writer_guid (rti.connexdds.SampleIdentity property)`, 508
- `writer_instance_cache_allocation` (*rti.connexdds.PersistentStorageSettings property*), 414
- `writer_loaned_sample_allocation` (*rti.connexdds.DataWriterResourceLimits property*), 134
- `writer_memory_state` (*rti.connexdds.PersistentStorageSettings property*), 414
- `writer_property_list_max_length` (*rti.connexdds.DomainParticipantResourceLimits property*), 175
- `writer_property_string_max_length` (*rti.connexdds.DomainParticipantResourceLimits property*), 175
- `writer_removed_batch_sample_dropped_sample_count` (*rti.connexdds.DataReaderCacheStatus property*), 91
- `writer_resource_limits` (*rti.connexdds.DataWriterQos property*), 132
- `writer_return_loan ()` (*rti.connexdds.DynamicData.WriterContentFilter method*), 220
- `writer_return_loan ()` (*rti.connexdds.ParticipantBuiltinTopicData.WriterContentFilter method*), 404
- `writer_return_loan ()` (*rti.connexdds.PublicationBuiltinTopicData.WriterContentFilter method*), 461
- `writer_return_loan ()` (*rti.connexdds.ServiceRequest.WriterContentFilter method*), 557
- `writer_return_loan ()` (*rti.connexdds.SubscriptionBuiltinTopicData.WriterContentFilter method*), 619
- `writer_return_loan ()` (*rti.connexdds.TopicBuiltinTopicData.WriterContentFilter method*), 674
- `writer_sample_cache_allocation` (*rti.connexdds.PersistentStorageSettings property*), 414
- `writer_side_filter_optimization` (*rti.connexdds.ExpressionProperty property*), 269
- `writer_user_data_max_length` (*rti.connexdds.DomainParticipantResourceLimits property*), 175

property), 175

WriterDataLifecycle (*class in rti.connextdds*), 756

WSTRING_TYPE (*rti.connextdds.TypeKind attribute*), 725

WSTRING_TYPE (*rti.connextdds.TypeKind.TypeKind attribute*),
723

WstringSeq (*class in rti.connextdds*), 756

WStringType (*class in rti.connextdds*), 747

X

XCDR (*rti.connextdds.DataRepresentation attribute*), 109

XCDR2 (*rti.connextdds.DataRepresentation attribute*), 109

XML (*rti.connextdds.DataRepresentation attribute*), 109

XML (*rti.connextdds.PrintFormatKind attribute*), 421

XML (*rti.connextdds.PrintFormatKind.PrintFormatKind attribute*),
421

xml (*rti.connextdds.PrintFormatProperty attribute*), 423

Z

ZERO (*rti.connextdds.CdrPaddingKind attribute*), 62

ZERO (*rti.connextdds.CdrPaddingKind.CdrPaddingKind
attribute*), 61

zero (*rti.connextdds.Duration attribute*), 184

zero (*rti.connextdds.SequenceNumber attribute*), 520

zero (*rti.connextdds.Time attribute*), 638

ZLIB (*rti.connextdds.CompressionIdMask attribute*), 70